# Temporale Aspekte entdeckten Wissens: Ein Bezugssystem für die Evolution von Mustern

D I S S E R T A T I O N

zur Erlangung des akademischen Grades
doctor rerum politicarum
(dr. rer. pol.)
im Fach Wirtschaftswissenschaft

eingereicht an der
Wirtschaftswissenschaftlichen Fakultät
Humboldt-Universität zu Berlin

von
Herrn Dipl.-Kfm. Steffan Baron
geboren am 21.11.1970 in Berlin

Präsident der Humboldt-Universität zu Berlin:
Prof. Dr. Jürgen Mlynek

Dekan der Wirtschaftswissenschaftlichen Fakultät:
Prof. Michael Burda, Ph.D.

Gutachter:

1. Prof. Oliver Günther Ph.D.
2. Prof. Dr. Myra Spiliopoulou

| | |
|---|---|
| eingereicht am: | 17. Februar 2004 |
| Tag der mündlichen Prüfung: | 18. Juni 2004 |

**Abstract**

Over the past few years the number and size of datasets available electronically have grown significantly. This has stimulated research into the development of efficient techniques for the discovery of valuable and utilisable knowledge in this data. Traditionally the emphasis has been on criteria such as performance and scalability; in recent years, however, the temporal dimension of the data has become a focus of interest. Methods have been developed that deal with maintaining and updating the discovered knowledge. These approaches are based on the assumption that the data is collected over a long period of time and, thus, affected by the same changes as the aspects of reality captured in the data. Hence, changes to the distribution or composition of the data will also be reflected in changes to the results of analysing the data. Therefore, it is not sufficient to consider only the non-temporal aspects of the knowledge used to support decision makers, rather it becomes a necessity to also consider the development of identified patterns over time. Then and only then, does it become possible to make a balanced decision.

In this work, knowledge discovery is considered to be a continuous process: data is collected over a period of time and analysed at specific time intervals. Each analysis produces a set of patterns which are stored in a rule base and monitored based on their statistical properties. Using a temporal data model which consists of both the content of a pattern, i.e., the relationship in the data the pattern reflects, and its statistical measurements which are used to estimate the quality of the discovered model, a general framework for monitoring and analysing the development of the discovered knowledge is proposed. Integrating the many different facets of pattern evolution, the model also provides for trend recognition. The framework is used to detect and assess different types of pattern change with respect to their qualitative, quantitative and temporal aspects. In addition, it permits the usage of the temporal properties of patterns as criterion for their relevance and enables the application expert to determine the causes of pattern change. Two case

studies are presented and discussed which examine the eligibility of the proposed concepts thoroughly.

**Keywords:**
Data Mining, Knowledge Discovery in Databases, Pattern Monitoring, Pattern Evolution

**Zusammenfassung**

In den vergangenen Jahren haben Anzahl und Umfang der elektronisch verfügbaren Datensätze stark zugenommen, wodurch die Entwicklung effizienter Methoden zur Entdeckung nützlichen und verwertbaren Wissens in den Daten zu einer großen Herausforderung geworden ist. Während sonst Kriterien wie Performanz und Skalierbarkeit im Vordergrund standen, wurde in jüngerer Zeit auch die temporale Dimension der Daten einbezogen. Es wurden Methoden erarbeitet, die der Erhaltung und Aktualisierung des entdeckten Wissens dienen. Diesen Techniken liegt die Idee zugrunde, daß Daten im allgemeinen über einen längeren Zeitraum gesammelt werden. Damit sind sie den gleichen Änderungen ausgesetzt wie der durch die Daten abgebildete Ausschnitt der Realität. Ändert sich die Verteilung oder die Zusammensetzung der Daten, ist auch mit Veränderungen in den Analyse-Ergebnissen zu rechnen. Sollen die gefundenen Zusammenhänge zur Unterstützung von Entscheidungsträgern verwendet werden, genügt es aber keineswegs, nur die Aktualität der Ergebnisse sicherzustellen. Vielmehr ist es notwendig, auch die Entwicklung der gefundenen Zusammenhänge im Zeitverlauf zu erfassen. Erst durch eine solche Betrachtung ist eine zielorientierte Bewertung der Ergebnisse möglich, die sich unmittelbar auf die zu treffenden Entscheidungen auswirkt.

In der vorliegenden Arbeit wird Wissensentdeckung deshalb als kontinuierlicher Prozeß verstanden. Daten werden über einen potentiell langen Zeitraum gesammelt und in bestimmten Zeitabständen analysiert. Jede Analyse liefert eine Menge von Mustern, die in einer Regelbasis erfaßt und deren Entwicklung aufgezeichnet wird. Ausgangspunkt ist ein temporales Datenmodell, das sowohl den Inhalt von Mustern – d. h. den Zusammenhang in den Daten, der durch das Muster beschrieben wird – als auch seine statistischen Eigenschaften, die Aussagen hinsichtlich der Qualität des gefundenen Modells erlauben, abbildet. Darauf aufbauend, wird ein umfassendes Bezugssystem für die Überwachung und Analyse der Entwicklung entdeckten Wissens entwickelt, das die vielen verschiedenen Facetten der Evolution von Mustern integriert und die Erkennung von Trends erlaubt. Dieses Bezugssystem ermöglicht es einerseits, verschiedene Arten von Musteränderungen nach qualitativen, quantitativen und temporalen Kriterien erkennen und bewerten zu können, andererseits gestattet es, die temporalen Eigenschaften

der gefundenen Zusammenhänge als Kriterium für ihre Relevanz zu nutzen und die Ursachen der beobachteten Änderungen zu bestimmen. Im Rahmen zweier Fallstudien wurden die vorgestellten Konzepte einer eingehenden Überprüfung unterzogen.

**Schlagwörter:**
Data Mining, Wissensentdeckung in Datenbanken, Musterüberwachung, Musterevolution

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Kapitel 1

# Einleitung

Die rasante technologische Entwicklung versetzt uns seit einigen Jahren in die Lage, jeden Tag kaum noch vorstellbare Datenmengen zu sammeln, und eine Reihe von Statistiken gibt Aufschluß darüber, mit welcher Rate die Menge der gespeicherten Informationen täglich bzw. jährlich wächst. So wird beispielsweise eine Studie, die der Frage nachgeht, wieviel Daten jährlich generiert werden, in regelmäßigen Abständen an der *University of California at Berkeley* durchgeführt.[1] In der aktuellen Version wird das Volumen an *neuen* Daten für das Jahr 1999 auf 2 bis 3 Exabyte, für das Jahr 2003 gar auf 5 Exabyte geschätzt. Während ein großer Teil dieser Daten quasi als Nebenprodukt automatischer Prozesse z. B. in Produktionsanlagen anfällt, wird der weitaus kleinere Teil der Daten zielgerichtet gesammelt. Ungeachtet der Quelle oder des Verwendungszwecks ist allen gesammelten Daten gemeinsam, daß sie zu einer bestimmten Zeit angefallen sind, entweder explizit, indem jeder Eintrag mit einem Zeitstempel versehen ist, oder implizit, weil die Einträge im Aufzeichnungsprotokoll einer bestimmten Periode enthalten sind. Obwohl die zeitliche Dimension oft sogar explizit als Teil des Datenmodells erfaßt ist, vernachlässigt man sie bei den Ergebnissen der auf Basis dieser Daten vorgenommenen Analysen fast vollständig.

In der Praxis gilt ein Datensatz für gewöhnlich als statische Einheit. Man läßt hierbei außer acht, daß die Ergebnisse der Auswertung wie die Daten selbst eine temporale Dimension aufweisen und daß sich ihre Eigenschaften im Zeitverlauf ändern können. So stellen Berry und Linoff [1997] zwar fest, daß sich die entdeckten Zusammenhänge ändern können, weil die aufgrund

[1] `http://www.sims.berkeley.edu/research/projects/how-much-info` (24.02.2004)

der Ergebnisse getroffenen Entscheidungen Auswirkungen auf die der Analyse zugrunde liegenden Daten haben können, sie schlagen jedoch lediglich vor, daß Data Mining ein *Zyklus* sein sollte (vgl. Abbildung 1.1). Nachdem die

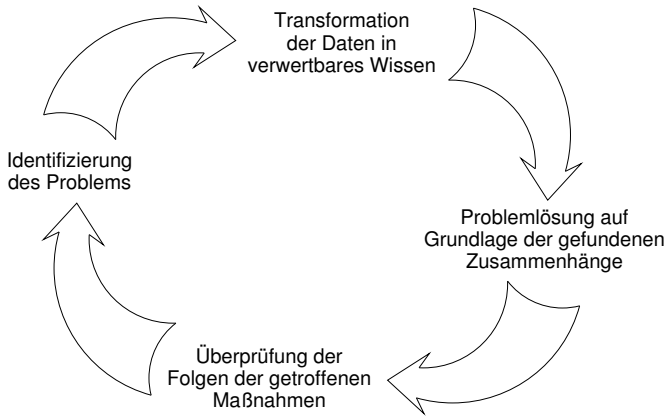


Abbildung 1.1: „*The Virtuous Cycle of Data Mining*“

Zielstellungen der Analyse festgelegt worden sind und geeignete Daten ausgewählt und analysiert wurden, werden die Auswirkungen der auf Basis der Ergebnisse getroffenen Entscheidungen überprüft, indem wieder von neuem Daten gesammelt und analysiert werden.

Diese Herangehensweise behebt das Problem jedoch nur partiell, da bloß ein Teil der möglichen Ursachen für die Änderungen der Muster betrachtet wird. Nicht nur die getroffenen Entscheidungen nehmen Einfluß auf die Daten, sondern auch andere, externe Faktoren. Hinzu kommt, daß diese externen Einflüsse auch eine zeitliche Dimension haben, die ebenfalls in die Analyse einbezogen werden sollte. Das bedeutet, daß der Einfluß externer Faktoren auf die Daten eventuell nur vorübergehend wirksam ist und sie die gefundenen Zusammenhänge lediglich für einen kurzen Zeitraum verändern. Natürlich kann auch der Fall eintreten, daß sich Einflußfaktoren langfristig oder periodisch ändern. Aber auch in diesem Fall sollten die Auswirkungen auf die Analyse-Ergebnisse die gleiche zeitliche Dimension aufweisen und – wenn auch mit einer gewissen Verzögerung – nachzuweisen sein.

Bei Anwendung der von Berry und Linoff empfohlenen Vorgehensweise ist diese Art der Information jedoch nicht zu ermitteln, da die Datenerfassung nicht fortlaufend erfolgt. Der Zyklus findet nur in diskreter Zeit statt,

wodurch die Entwicklung der gefundenen Zusammenhänge im Zeitverlauf vollständig verlorengeht. Nur wenn der Zyklus kontinuierlich durchlaufen wird, kann ermittelt werden, ob und – wenn ja – wie sich das entdeckte Wissen ändert, wann es sich verändert und – wenn möglich – warum es sich geändert hat.

## 1.1 Motivation

Das folgende Beispiel soll die grundlegende Problematik skizzieren, für die im Rahmen dieser Arbeit Lösungen erarbeitet worden sind.

**Beispiel 1.1** Wir betrachten einen Wissenschaftlichen Mitarbeiter, der an seiner Dissertation arbeitet. Während er zu diesem Zweck für gewöhnlich die gesamte Arbeitszeit am Institut verbringt, schreibt er in der „heißen Phase“ aus naheliegenden Gründen lieber zu Hause. Mit der Abgabe seiner Dissertationsschrift – die hoffentlich angenommen wird – kehrt er zu seinem alten Rhythmus zurück, am Institut zu arbeiten. Ein halbes Jahr später endet sein Arbeitsvertrag mit der Universität, und bis er einen gut dotierten Posten bei einem großen Unternehmen gefunden hat, arbeitet er wieder daheim.

Die wechselnden Lebensumstände beeinflussen dabei unter anderem seine häusliche Internet-Nutzung. So stellt sich aus der Sicht seines *Internet Service Providers* (ISP) der beschriebene Sachverhalt wie folgt dar: Nach einem langen Zeitraum mit vergleichsweise niedrigen Umsätzen folgt eine kurze Zeit, in der unser Wissenschaftlicher Mitarbeiter sehr oft und lange *online* ist, gefolgt von einem Zeitraum, in dem sich das Ausmaß der Internet-Nutzung wieder auf seinem alten Niveau einpendelt. Anschließend steigt die Nutzung wieder stark an, um sich schließlich auf dem neuen, höheren Niveau zu stabilisieren. ◇

In diesem Beispiel lassen sich drei unterschiedliche Typen von Änderungen erkennen – erst die Abweichung vom Normalniveau, dann die Rückkehr zum ursprünglichen Niveau und schließlich die dauerhafte Änderung auf ein neues, möglicherweise permanent höheres Niveau. Man könnte jedoch die beiden ersten Änderungen auch als eine einzelne, kurzfristige Änderung betrachten und die letzte Änderung als langfristig ansehen. Für den ISP sind beide Änderungen interessant, obwohl es aufgrund der ihm zur Verfügung stehenden Daten keine Korrelation zwischen beiden gibt, da er die Lebensumstände seines Kunden nicht kennt. Bei einem einzelnen Individuum ließen

sich die beschriebenen Änderungen sicher nur schwer bestimmen und würden wohl kaum ins Gewicht fallen. Betrachtet man jedoch alle Studenten, die zum Kundenkreis des ISP gehören, und den Wechsel von Vorlesungszeit und Semesterferien, sind die Änderungen ohne weiteres zu ermitteln und bestimmt auch von wirtschaftlicher Tragweite.

Es wird deutlich, daß es Einflußfaktoren geben kann, die sich zwar in den gesammelten Daten niederschlagen, die jedoch nicht selbst *Bestandteil* der Daten sind, z. B. über ein entsprechendes Attribut. Während sehr viele Einflußfaktoren – vielleicht sogar der größere Teil – nicht direkt identifiziert werden können, sind sie in den meisten Fällen relativ klar an ihren Wirkungen zu erkennen. Allen diesen Faktoren ist gemeinsam, daß sie eine zeitliche Dimension aufweisen, d. h., der Beginn und das Ende ihres Einflusses lassen sich bestimmten Zeitpunkten zuordnen. Das Problem liegt jedoch darin, daß man diese Zeitpunkte nicht ohne weiteres bestimmen kann, da die Faktoren annahmegemäß nicht erfaßt und deshalb unbekannt sind. Es ist allerdings möglich, diese Zeitpunkte anhand des Einflusses der Faktoren auf die Daten zu ermitteln. Mit diesem Wissen sollte es dann in der Folge gelingen, Änderungen in den Daten auf ihre auslösenden Faktoren zurückzuführen. Unabdingbare Voraussetzung dafür ist jedoch, daß die zeitliche Dimension nicht vernachlässigt wird.

Ein weiteres Problem ist in der Bewertung der *Wichtigkeit* der gefundenen Muster zu sehen. Bisher werden dabei weitestgehend konventionelle Maßstäbe angelegt. Das bedeutet, daß im wesentlichen die statistischen Eigenschaften der Muster verwendet werden, um ihre Bedeutung für den Benutzer einzuschätzen. Abgesehen vom Anwendungskontext gibt es jedoch auch eine andere Möglichkeit, das potentielle Interesse des Nutzers einzuschätzen: Oft sind die gefundenen Muster nicht nur interessant aufgrund ihrer konventionellen Eigenschaften wie Support, Konfidenz usw., sondern vielleicht auch oder gerade *wegen* der Änderungen, die sie über die Zeit hin erfahren. Insbesondere wenn der Nutzer mit den gefundenen Ergebnissen bereits vertraut ist, werden die Änderungen, denen die Ergebnisse im Zeitverlauf unterliegen, von besonderem Interesse für ihn sein [Chakrabarti u. a. 1998].

Welche Betrachtungsweise auch im Vordergrund steht, es wird deutlich, daß die Kenntnis und Beachtung der temporalen Eigenschaften von Mining-Ergebnissen zusätzliche, wertvolle Einblicke liefern kann, da die Unterstützung von Entscheidungsprozessen ein *kontinuierlicher* Vorgang ist, also eine laufende Überprüfung getroffener Entscheidungen erfordert. Kann aber die Entscheidungsgrundlage nicht als Konstante betrachtet werden, so muß

natürlich auch sie fortlaufend überprüft werden. Dazu ist es zunächst notwendig, die Ergebnisse in einer Weise darzustellen, die eine Erfassung ihrer Änderungen ermöglicht und diese bestimmten Zeitpunkten zuordnet. Das Muster muß dabei in seiner *Gesamtheit* betrachtet werden, d. h., sowohl der Zusammenhang in den Daten, den das Muster beschreibt, als auch die statistischen Eigenschaften des Musters, die sich im Zeitverlauf ändern können, müssen erfaßt werden. Auf der Grundlage eines solchen Modells lassen sich dann Untersuchungen darüber vornehmen, welchen Änderungen ein Muster im Zeitverlauf unterworfen ist. Sollen bestimmte Arten von Änderungen unterschieden werden, ist es zudem notwendig, Kriterien festzulegen, die ihre Identifizierung ermöglichen. Dabei muß auch geklärt werden, welche Dimensionen die Musteränderungen aufweisen können.

Sind die Dimensionen der Muster und ihrer Änderungen bekannt, können Kriterien für die *Beurteilung* der Änderungen entwickelt werden, d. h. für die Abschätzung des Interesses, das ein Nutzer diesen Änderungen entgegenbringt. Im Normalfall ist die Anzahl der entdeckten Regeln selbst bei vergleichsweise kleinen Datensätzen schon relativ hoch, und bereits hier stellt sich die Frage, nach welchen Kriterien die Ergebnisse gefiltert werden sollen, um dem Anwender nur die wirklich *interessanten* Muster zu präsentieren. Das gleiche gilt auch für die Änderungen der Muster, und zwar in verschärfter Form, da die Zeit als zusätzliche Dimension in Betracht gezogen werden muß und die Anzahl der Änderungen sehr viel größer sein wird als die ursprüngliche Anzahl an Regeln. Ein weiterer Aspekt betrifft die Möglichkeit, daß sich das Interesse des Anwenders im Zeitverlauf ändert. Die Auswahl der Muster sollte deshalb nicht statisch erfolgen, sondern im Rahmen des Anwendungskontextes regelmäßig hinterfragt werden. Aber selbst wenn alle interessanten Regeländerungen identifiziert und bestimmten Zeitpunkten zugeordnet worden sind, wird der Nutzer nicht ohne weiteres die externen Faktoren bestimmen können, welche die Änderungen verursacht haben. Deswegen ist es wichtig, daß zumindest jene Ursachen ermittelt werden, die aus dem Datensatz hervorgehen. Oft tritt der Fall ein, daß Muster inhaltliche Ähnlichkeiten aufweisen, weil sie die gleichen Bestandteile haben. Ändern sich diese Muster in vergleichbarem Maße, ist es wahrscheinlich, daß die Änderungen jeweils auf die gleichen Ursachen zurückgeführt werden können. In einem solchen Fall sollten dem Nutzer nicht nur die Musteränderungen angezeigt werden, sondern auch ihre Ursache. Je spezifischer der Anwender informiert werden kann, desto größer ist für ihn die Chance, einen Zusammenhang zwischen den Änderungen in den Daten und den externen Faktoren, die nicht im Datensatz

erfaßt sind, herzustellen.

## 1.2 Aufbau der Dissertation

Für die meisten der angesprochenen Problemfelder wurden im Rahmen dieser Arbeit Lösungen entwickelt. Ziel war es, die vielen verschiedenen Facetten der Evolution von Mustern in einem einheitlichen Bezugssystem zu erfassen, um einerseits Musteränderungen erkennen und bewerten zu können und andererseits die temporalen Eigenschaften eines Musters als Kriterium für die *Relevanz* von Mustern zu nutzen. Grundlage dieses Bezugssystems bildet ein Datenmodell, das in der Lage ist, sowohl den Inhalt von Mustern als auch die zeitliche Entwicklung ihrer statistischen Eigenschaften darzustellen. Nach dem im Kapitel 3 gegebenen Literaturüberblick wird im Kapitel 4 ein temporales Regelmodell vorgestellt, das diesen Anforderungen gerecht wird und in seiner ursprünglichen Form in [Baron und Spiliopoulou 2001] beschrieben wurde. Es besteht aus einem invarianten Teil, der den Inhalt des Musters speichert, und einem variablen Teil, in dem die statistischen Eigenschaften des Musters abgelegt sind, welche von Periode zu Periode fortgeschrieben werden. Dieses Datenmodell und seine Transformation in ein relationales Datenmodell bilden die Grundlage für die Verwaltung und Aktualisierung der entdeckten Muster in einem relationalen Datenbanksystem.

Kapitel 5 stellt die Techniken vor, die für die Erkennung und Bewertung verschiedener Arten von Musteränderungen nach qualitativen und quantitativen Kriterien verwendet werden und die in [Baron u. a. 2003] und [Baron und Spiliopoulou 2003] veröffentlicht wurden. Einerseits werden Methoden entwickelt, um Änderungen erkennen zu können, andererseits geht es um die Frage, wie die Änderungen bewertet werden sollen und wie sie als Grundlage zur Einschätzung der *Wichtigkeit* von Mustern verwendet werden können. Zusätzlich kann mit Hilfe der vorgeschlagenen Methoden eine Abschätzung der temporalen Dimension der beobachteten Änderungen vorgenommen werden, und bei Bedarf können die Ursachen bestimmt werden, die für die Veränderungen verantwortlich sind und aus dem erfaßten Datensatz hervorgehen.

Das in den Kapiteln 4 und 5 entwickelte Bezugssystem dient als Grundlage für den im Kapitel 6 vorgestellten *Pattern Monitor* (PAM). Gemäß dem entwickelten temporalen Regelmodell implementiert der Monitor die verschiedenen Methoden zur Erkennung und Analyse von Musteränderun-

gen. Er definiert einen Prozeß, in dem iterativ Daten analysiert und die in der Regelbasis gespeicherten Muster aktualisiert und fortgeschrieben werden. In einem zweiten Schritt werden dann die Änderungen in der Regelbasis analysiert und bewertet. Im Gegensatz zu konventionellen Werkzeugen der Wissensentdeckung besteht das Ergebnis der Analyse nicht nur aus den entdeckten Mustern, sondern zusätzlich auch aus ihren Änderungen und ihrer Bewertung. Im letzten Schritt der Analyse wird versucht, die Ursachen für die Änderungen zu bestimmen, soweit sie aus dem untersuchten Datensatz hervorgehen. Zu diesem Zweck wird für jede Musteränderung bestimmt, durch welche anderen Änderungen sie ausgelöst wurde bzw. welche Änderungen dazu beigetragen haben [Baron und Spiliopoulou 2003].

Das folgende Kapitel ist dazu gedacht, einen kurzen Einblick in den Bereich Data Mining zu geben. Natürlich ist es an dieser Stelle unmöglich und auch nicht gewollt, einen umfassenden Überblick über dieses Gebiet zu vermitteln. Dazu sei hier auf die heute in großem Umfang verfügbare Standardliteratur verwiesen (vgl. z. B. Berry und Linoff [1997], Witten und Frank [1999] sowie Hand und Mannila [2001]). Es sollen jedoch einige grundlegende Konzepte beschrieben werden, welche die Basis für die nachfolgenden Kapitel bilden. Neben den verschiedenen Methoden des Data Mining und ein paar weitverbreiteten Qualitätsmaßen, die häufig zur Beurteilung der gefundenen Ergebnisse herangezogen werden, sollen hier vor allem die verschiedenen Arten von Mining-Ergebnissen beschrieben werden.

# Kapitel 2

# Grundlagen

## 2.1 Einleitung

Nach über zehn Jahren aktiver Forschung gibt es eine ganze Reihe von Versuchen, eine grundlegende Begriffsbestimmung durchzuführen.[1] Eine weithin anerkannte Definition bezeichnet Data Mining als den automatischen oder semiautomatischen Prozeß der Entdeckung von potentiell nützlichem, zuvor unbekanntem Wissen in sehr großen Datensätzen [Frawley u. a. 1992]. Häufig wird synonym auch der Begriff *Knowledge Discovery in Databases* (KDD) verwendet, insbesondere wenn die zu analysierenden Daten in einem Datenbanksystem vorgehalten werden.[2] Eine andere Sichtweise bezeichnet dagegen den Prozeß selbst als Knowledge Discovery und sieht Data Mining nur als den Teil davon, in dem die Extraktion des Wissens erfolgt. Ungeachtet der verwendeten Definition stellt Data Mining jedoch weniger eine separate Disziplin dar, sondern kennzeichnet eher eine Sammlung von Methoden und Techniken aus verschiedenen Bereichen der Informatik, wie z. B. dem Maschinellen Lernen und der Künstlichen Intelligenz, und der Mathematik und Statistik. Darüber hinaus hat Data Mining im allgemeinen einen sehr starken Praxisbezug, d. h., die Ergebnisse sollten für den Anwender möglichst verständlich und mit wenig Aufwand in eine Lösung für das jeweilige Problem umzuset-

[1] Viele der Methoden, die heute unter dem Begriff Data Mining zusammengefaßt werden, sind jedoch deutlich älteren Datums.

[2] Die Definition im Artikel von Frawley u. a. [1992] bezieht sich eigentlich auf *Knowledge Discovery in Databases* – der Begriff *Data Mining* wird lediglich am Rande erwähnt. Während die Bezeichnung *Knowledge Discovery in Databases* verstärkt im akademischen Kontext Verwendung findet, ist in der Industrie häufiger von *Data Mining* die Rede.

zen sein. Die Methoden, die unter dem Begriff Data Mining zusammengefaßt werden, finden inzwischen in einer Vielzahl verschiedener wissenschaftlicher Disziplinen Anwendung, z. B. in der Astronomie, Physik, Biologie, Medizin und Chemie. Sie werden aber auch in der Industrie eingesetzt, insbesondere in den Bereichen Finanzwirtschaft, Absatz, Marketing und Kundenpflege.

Läßt man einmal die jeweilige Anwendungsdomäne außer Betracht, so verläuft der *Prozeß* der Wissensentdeckung im allgemeinen sehr ähnlich: Nach einem in den meisten Fällen sehr aufwendigen Schritt zur Aufbereitung der Daten wird ein Modell abgeleitet, das die Beziehungen in den Daten möglichst gut beschreibt und das für Vorhersagen bezüglich der erwarteten Entwicklung der betrachteten Anwendungsdomäne verwendet werden kann. Die Möglichkeit zur Vorhersage stellt dabei jedoch nicht immer eine der primären Zielstellungen dar. Oft kommt es lediglich darauf an, ein Modell zu finden, um die in den Daten enthaltenen Beobachtungen des betrachteten Ausschnitts der Realität möglichst gut zu erklären. Eine sehr viel wichtigere Rolle spielt die Vorhersage jedoch für die Methoden, die dem Maschinellen Lernen entliehen sind. Allgemein formuliert, wird dabei eine Menge von bekannten Instanzen als Eingabe für einen Lernprozeß verwendet. Das daraus resultierende bzw. *gelernte* Modell wird dann eingesetzt, um Aussagen über eine Menge unbekannter Instanzen zu machen. Das Prinzip der Vorhersage bezieht sich dabei nicht auf die Zukunft, sondern auf einen unbekannten Aspekt des betrachteten Ausschnitts der Realität.

Berry und Linoff [1997] fassen den Prozeß des Data Mining noch etwas weiter. Indem sie den starken Praxisbezug in den Vordergrund stellen, beginnt für sie der Prozeß bereits mit der Identifizierung der anwendungsabhängigen Problemstellung. Sie bildet den Startpunkt eines Zyklus, in dessen Verlauf die für eine mögliche Lösung geeignet erscheinenden Daten gesammelt, analysiert und die Ergebnisse interpretiert werden. Während dieser Schritt bzw. die Verwendung des gefundenen Modells zur Vorhersage im Normalfall dem Ende des Prozesses entspricht, beschließt für Berry und Linoff erst die Überprüfung der Auswirkungen der auf Basis der interpretierten Mining-Ergebnisse gefällten Entscheidungen den Zyklus (vgl. Abbildung 1.1).

Im Rahmen dieser Arbeit wird dieser Prozeß nochmals erweitert, indem zusätzlich zu den bereits erwähnten Auswirkungen eigener, auf Basis früherer Ergebnisse getroffener Entscheidungen auch die potentiellen Auswirkungen aller anderen Einflüsse betrachtet werden. Gemäß dieser Betrachtungsweise müssen die gefundenen Zusammenhänge nicht nur zur Überprüfung des Erfolgs der ergriffenen Maßnahmen aktualisiert werden, sondern auch, um die

Auswirkungen sich ändernder, externer Einflüsse zu berücksichtigen. Im Ergebnis werden die entdeckten Zusammenhänge einer kontinuierlichen Überprüfung unterzogen und ihre zeitliche Entwicklung aufgezeichnet, wobei die so erhaltenen Daten den Ausgangspunkt für weitere Analysen bilden.

## 2.2 Mining-Techniken

Wie im vorhergehenden Abschnitt beschrieben wurde, verwenden Methoden des Maschinellen Lernens bekannte Instanzen, um ein Modell abzuleiten, das Aussagen über unbekannte Instanzen erlaubt. Dieses stark vereinfachte Schema – im allgemeinen mit *Supervised Learning* bezeichnet – stellt jedoch nur eine der beiden grundlegenden Vorgehensweisen dar. Beim *Unsupervised Learning* hingegen wird auch das Modell auf Grundlage der *unbekannten* Instanzen bestimmt.[3] Ein weiterer wesentlicher Unterschied zwischen beiden Vorgehensweisen ist hinsichtlich ihrer Zielstellung zu erkennen. Während es im ersten Fall zumeist um die Erklärung bzw. Vorhersage einer Zielvariablen geht, wird im zweiten Fall nach allen Zusammenhängen gesucht, die im betrachteten Datensatz eine bestimmte, vordefinierte Evidenz aufweisen.

Der Unterschied zwischen beiden Vorgehensweisen läßt sich am besten anhand der Gegenüberstellung von Klassifikation und Cluster-Analyse verdeutlichen: Bei der Klassifikation, einem typischen Vertreter für das Supervised Learning, werden die Objekte eines *vorklassifizierten* Datensatzes verwendet, um ein Modell zu lernen, den sogenannten Klassifikator (engl. *Classifier*). Die Menge der Klassen ist dabei entweder explizit vorgegeben oder implizit durch die Menge der verschiedenen Klassen in den vorklassifizierten Daten festgelegt. Das gelernte Modell wird dann verwendet, um eine Zuordnung bislang unbekannter Objekte zu einer der Klassen vorzunehmen. Die Zielvariable in diesem Prozeß ist somit die Klassenzugehörigkeit der Objekte.[4] Die Cluster-Analyse arbeitet dagegen mit *unklassifizierten* Daten und versucht, unter Verwendung eines festgelegten Abstandsmaßes Gruppen ähnlicher Objekte zu entdecken, denen diese dann zugeordnet werden können. Prinzipiell sind diese Gruppen mit den für die Klassifikation verwendeten Klassen vergleich-

[3] Zum Teil werden die beiden Vorgehensweisen auch als *Directed* und *Undirected Data Mining* bezeichnet [Berry und Linoff 1997].

[4] Dies ist jedoch nur eine vereinfachte Darstellung. In der Realität würde das gelernte Modell zunächst noch getestet und, wenn möglich, weiter verbessert, bevor es zur Anwendung käme.

bar, sie sind jedoch *nicht* vordefiniert, sondern werden erst im Rahmen der Analyse bestimmt. Während im ersten Fall die Zielstellung somit in der Zuordnung der unbekannten Instanzen zu einer der verschiedenen Klassen zu sehen ist, geht es im zweiten Fall primär um die Bestimmung der Klassen selbst.

Diese grundlegende Unterscheidung von Mining-Paradigmen ließe sich natürlich weiter verfeinern. Prinzipiell unterscheiden sich die verschiedenen Methoden einerseits im Hinblick auf die Zielstellungen, die sie verfolgen, andererseits jedoch auch durch die Form der zu erwartenden Ergebnisse. Für die vorliegende Arbeit sind die Methoden des Unsupervised Learning besonders wichtig, und im folgenden sollen drei wichtige Vertreter, die insbesondere für das nächste Kapitel von Interesse sind, detaillierter dargestellt werden.

### 2.2.1 Warenkorbanalyse

Allgemein formuliert, wird im Rahmen der Warenkorbanalyse versucht, Beziehungen zwischen Objekten zu finden, die den für den betrachteten Datensatz festgelegten Kriterien in puncto Häufigkeit bzw. Evidenz ihres Auftretens genügen. Dazu werden Transaktionen betrachtet, die jeweils ein oder mehrere Objekte enthalten. Eine Transaktion kann dabei als logische Arbeitseinheit gesehen werden und wird über eine eindeutige Transaktionskennung referenziert, während die Objekte als elementare Entitäten betrachtet werden können, die in der jeweiligen Transaktion zusammengefaßt sind bzw. gemeinsam auftreten. Die gebräuchlichste Anwendung dieser Methode, von der auch ihr Name abgeleitet wurde, ist die Analyse der Verkäufe in einem Supermarkt. Jeder Kaufakt eines Kunden entspricht dabei einer Transaktion, in der ein oder mehrere Produkte erworben werden.

Formal ausgedrückt, wird eine Menge von Literalen $I = \{i_1, i_2, ..., i_n\}$ betrachtet, die als *Items* bezeichnet werden, und eine Datenbank $D$, die eine Menge von Transaktionen $T$ enthält. Jede Transaktion enthält eine Menge von Items $X \subseteq I$. Eine Assoziationsregel ist dann eine Implikation der Form $X \Rightarrow Y$, wobei $X \subset I$, $Y \subset I$ und $X \cap Y = \emptyset$ gilt [Agrawal u. a. 1993]. Die linke Seite (engl. *left hand side*, abgekürzt *lhs*) bzw. Voraussetzung einer Assoziation wird dabei häufig als *Body* der Regel bezeichnet, die rechte Seite (engl. *right hand side*, abgekürzt *rhs*) bzw. das Ergebnis der Assoziation als *Head* der Regel. Das Ziel der Analyse besteht darin, Assoziationsregeln zu finden, die vordefinierte Grenzen der Konfidenz $c$ und der relativen Häufigkeit ihres Auftretens, die als Support $s$ bezeichnet wird, erreichen bzw. überschrei-

ten. Eine Regel gilt mit Konfidenz $c$, wenn $c\%$ der Transaktionen, welche die linke Seite der Assoziation enthalten, auch ihre rechte Seite enthalten, und mit Support $s$, wenn $s\%$ der Transaktionen in $D$ Body *und* Head der Regel enthalten. Als Parameter für die Analyse gibt der Nutzer die Schwellen für die Konfidenz $\tau_c$ und den Support $\tau_s$ an, und das Ergebnis umfaßt alle Assoziationen, die diesen Kriterien genügen.[5]

Für die Lösung dieser Problemstellung wurde im Laufe der Jahre eine Vielzahl von Algorithmen vorgeschlagen. Einer der bekanntesten ist der Algorithmus *Apriori* [Agrawal und Srikant 1994]. Er folgt einem sehr simplen, aber effizienten Prinzip: In einer Reihe von Iterationen werden zunächst all jene Mengen von Items bestimmt, die gemäß dem Kriterium $\tau_s$ *häufig* sind, d. h., der Anteil der Transaktionen in $D$, die sie enthalten, liegt mindestens bei $\tau_s$. Diese Mengen werden als *Frequent Itemsets* bezeichnet. In jeder Iteration wird dabei die Anzahl der Items basierend auf den häufigen Itemsets der letzten Iteration um 1 erhöht, so daß in Iteration $k$ gerade $k$ Items in einem Itemset enthalten sind.[6] Dieser Vorgehensweise liegt die Annahme zugrunde, daß ein Itemset nur dann häufig sein kann, wenn alle darin enthaltenen Itemsets ebenfalls häufig sind. Im zweiten Schritt der Analyse werden aus den Elementen dieser Itemsets Assoziationen gebildet, die dem Kriterium $\tau_c$ genügen. Beispiel 2.1 soll die grundlegende Arbeitsweise von *Apriori* verdeutlichen.

**Beispiel 2.1** Es wird die in Tabelle 2.1 dargestellte Transaktionsdatenbank betrachtet. Die erste Spalte enthält die Kennungen der verschiedenen Trans-

| TID | Items |
|---|---|
| 1 | $\{A, B\}$ |
| 2 | $\{A, C, D\}$ |
| 3 | $\{A, E\}$ |
| 4 | $\{A, B, E\}$ |
| 5 | $\{B, D\}$ |

Tabelle 2.1: Beispiel einer Transaktionsdatenbank

[5] Die Grenzen $\tau_c$ und $\tau_s$ werden als einfacher Anteil oder, alternativ, als Prozentsatz angegeben.

[6] Im allgemeinen werden die Itemsets in den verschiedenen Iterationen deshalb auch als $k$-Itemsets bezeichnet.

aktionen, die zweite Spalte die Produkte bzw. Items, die in der jeweiligen Transaktion enthalten sind. Der Nutzer ist an allen Assoziationsregeln interessiert, die einen minimalen Support von $\tau_s = 0{,}25$ und eine minimale Konfidenz von $\tau_c = 0{,}75$ aufweisen.

In der ersten Iteration ermittelt *Apriori* zunächst alle Itemsets, die ein einzelnes Produkt enthalten und dem festgelegten Support-Kriterium genügen. Gemäß Tabelle 2.2 sind dies die Produkte $A$, $B$, $D$ und $E$. Basierend auf den

| Iteration | Itemset | Support | $\geq \tau_s$ |
|---|---|---|---|
| 1 | $\{A\}$ | 0,8 | $\surd$ |
| | $\{B\}$ | 0,6 | $\surd$ |
| | $\{C\}$ | 0,2 | – |
| | $\{D\}$ | 0,4 | $\surd$ |
| | $\{E\}$ | 0,4 | $\surd$ |
| 2 | $\{A, B\}$ | 0,4 | $\surd$ |
| | $\{A, D\}$ | 0,2 | – |
| | $\{A, E\}$ | 0,4 | $\surd$ |
| | $\{B, D\}$ | 0,2 | – |
| | $\{B, E\}$ | 0,2 | – |
| | $\{D, E\}$ | 0 | – |
| 3 | $\{A, B, E\}$ | 0,2 | – |

Tabelle 2.2: Bestimmung häufiger Itemsets

Ergebnissen der ersten Iteration, wird in der zweiten Iteration der Support aller Kombinationen dieser Itemsets bestimmt, die zwei Produkte enthalten. Bis auf $\{A, B\}$ und $\{A, E\}$ erfüllt kein Itemset die vorgegebene Support-Anforderung. In der dritten Iteration wird der Support aller Itemsets überprüft, die drei Produkte enthalten. Auch dieser Schritt beruht auf den Ergebnissen der vorangegangenen Iteration, so daß lediglich das Itemset $\{A, B, E\}$ überprüft werden muß. Da es kein häufiges Itemset mit 3 Produkten gibt, kann es auch keine häufigen Itemsets mit mehr als 3 Produkten geben, weswegen der erste Teil des Algorithmus terminiert.

Im zweiten Teil werden aus den häufigen Itemsets alle möglichen Assoziationsregeln gebildet und – sofern ihre Konfidenz mindestens $\tau_c = 0{,}75$ beträgt – ausgegeben. Für die betrachteten Daten sind dies die Itemsets $\{A, B\}$ und $\{A, E\}$. Tabelle 2.3 zeigt die gebildeten Regeln und ihre jeweilige Konfidenz.

Es ist ersichtlich, daß nur die Regel $E \Rightarrow A$ dem Konfidenz-Kriterium genügt,

| Assoziationsregel | Konfidenz | $\geq \tau_c$ |
|---|---|---|
| $A \Rightarrow B$ | 0,50 | – |
| $B \Rightarrow A$ | 0,67 | – |
| $A \Rightarrow E$ | 0,50 | – |
| $E \Rightarrow A$ | 1,00 | $\surd$ |

Tabelle 2.3: Bestimmung der Assoziationsregeln

alle anderen Regeln weisen Werte auf, die kleiner als 0,75 sind. ◇

Im Laufe der Jahre wurde eine Vielzahl von Änderungen vorgeschlagen, um die Effizienz von *Apriori* zu erhöhen; insbesondere ging es darum, die Anzahl der vergleichsweise teuren Lesezugriffe auf die Datenquelle zu vermindern. Es gibt jedoch auch eine Reihe anderer Algorithmen zur Entdeckung von Assoziationsregeln, und selbst heute, zehn Jahre nachdem *Apriori* vorgestellt wurde, ist die Lösung dieser Problemstellung noch Gegenstand aktiver Forschung.

### 2.2.2 Sequenzanalyse

Obwohl nur etwa zwei Jahre nach der Warenkorbanalyse zum ersten Mal erwähnt, ist die Anzahl der Arbeiten im Bereich der Sequenzanalyse deutlich kleiner als im Bereich der Warenkorbanalyse. In der ursprünglich vorgestellten Form besteht das Ziel der Analyse im Auffinden häufiger Sequenzen in einer potentiell großen Sequenzdatenbank, wobei eine Sequenz eine geordnete Menge von Transaktionen darstellt [Agrawal und Srikant 1995]. Wie schon bei der Warenkorbanalyse werden auch in diesem Fall Transaktionen betrachtet, die jeweils eine nichtleere Menge von Objekten enthalten. Abweichend werden hier jedoch das Individuum, das die Transaktion initiiert hat, und die zeitliche Ordnung der Transaktionen als zusätzliche Dimensionen einbezogen.

Formal ausgedrückt, wird eine Menge von Literalen $I = \{i_1, i_2, ..., i_n\}$ betrachtet, die als *Items* bezeichnet werden, und eine Datenbank $D$, die eine Menge von zeitlich geordneten Transaktionen $T$ enthält. Jede Transaktion umfaßt eine Menge Items $(i_1, i_2, \ldots, i_m)$, wobei jedes Item durch eine ganze

Zahl dargestellt wird. Eine Sequenz $\mathsf{s}$ ist eine geordnete Liste von Transaktionen und wird durch $\langle s_1, s_2, \ldots, s_n \rangle$ dargestellt, wobei $s_j$ ein Itemset ist. Eine Sequenz $\langle a_1, a_2, \ldots, a_n \rangle$ ist in einer anderen Sequenz $\langle b_1, b_2, \ldots, b_m \rangle$ enthalten, wenn es eine Menge ganzer Zahlen $i_1 < i_2 < \ldots < i_n$ gibt, so daß $a_1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, \ldots, a_n \subseteq b_{i_n}$ gilt. Eine Sequenz $\mathsf{s}$ ist maximal in einer Menge von Sequenzen $S$, wenn sie in keiner anderen Sequenz $\mathsf{s}'$ in $S$ enthalten ist. Eine Sequenz wird als häufig bezeichnet, wenn sie maximal ist und die vom Nutzer vorgegebene Grenze für den Support $\tau_s$ erreicht [Agrawal und Srikant 1995]. Der Support wird im Unterschied zur Warenkorbanalyse jedoch nicht in bezug auf die Gesamtzahl der Transaktionen bestimmt, sondern bezogen auf die Gesamtzahl der Individuen in der Transaktionsdatenbank $D$. Das bedeutet, daß der Support einer Sequenz $\mathsf{s}$ gleich dem Anteil der Individuen an der Gesamtzahl der Individuen in $D$ ist, deren Transaktionshistorie $\mathsf{s}$ enthält.

Eine allgemeinere Darstellung dieses Problems ist in [Mannila u. a. 1997] zu finden. Im vorgestellten Modell werden keine Kundentransaktionen betrachtet, sondern Sequenzen *generischer Ereignisse*, wobei neben der zeitlichen Ordnung der Ereignisse auch der *Abstand* zwischen den Ereignissen einbezogen wird.[7]

Sei $E$ die Menge der möglichen Ereignisse, dann ist eine Ereignissequenz $\mathsf{s}$ ein Tripel der Form $(s, T_S, T_E)$, wobei $s = \langle (A_1, t_1), (A_2, t_2), \ldots, (A_n, t_n) \rangle$, $A_j \in E$, und $T_S \leq t_j \leq T_E$ gilt. $T_S$ bezeichnet dabei den Zeitpunkt des Beginns der Sequenz, und $T_E$ markiert den Zeitpunkt des Endes der Sequenz. Als *Episode* bezeichnen Mannila u. a. eine partiell geordnete Menge von Ereignissen $A_j \subseteq E$, die als gerichtete azyklische Graphen dargestellt werden. Sie unterscheiden serielle Episoden, parallele und Episoden, die weder seriell noch parallel sind. Der Abstand zwischen den verschiedenen Ereignissen einer Sequenz wird mit Hilfe eines Zeitfensters ausgewertet, welches den maximalen Abstand zwischen zwei Ereignissen festlegt. Neben der Breite des Fensters gibt der Nutzer an, in wie vielen Zeitfenstern eine gegebene Episode enthalten sein muß, um als *häufig* betrachtet zu werden, d. h., die relative Häufigkeit bzw. der Support einer Episode ergibt sich als der Anteil jener Zeitfenster, in denen die Episode auftritt. Auch hier besteht das Ziel der Analyse in der Entdeckung häufiger Sequenzen. Im Gegensatz zu dem in [Agrawal und Srikant 1995] dargestellten Ansatz werden jedoch für den Fall, daß eine Episode $\beta$ in einer Episode $\gamma$ enthalten ist, formal dargestellt durch $\beta \preceq \gamma$, auch Regeln der Form $\beta \Rightarrow \gamma$ abgeleitet, deren Konfidenz sich analog

[7] Ein Ereignis kann dabei natürlich auch eine Kundentransaktion sein.

zur Warenkorbanalyse aus $s(\gamma)/s(\beta)$ ergibt.[8]

Obwohl auch zur Lösung dieses Problems viele verschiedene Algorithmen entwickelt wurden, läßt sich – vorausgesetzt, das in [Mannila u. a. 1997] dargestellte Modell liegt zugrunde – auch hier das Prinzip des *Apriori*-Algorithmus anwenden, d. h., in einer ersten Phase wird zunächst nach allen Sequenzen gesucht, die dem Kriterium $\tau_s$ genügen, und in der zweiten Phase werden aus diesen Sequenzen Regeln gebildet, die die Bedingung $\tau_c$ erfüllen. Um die relativ hohe Komplexität der Suche zu verringern, wurden aber auch Lösungsansätze für andere Problemstellungen adaptiert. So werden z. B. Indexstrukturen für Zeichenketten verwendet, die ursprünglich im *Information Retrieval* und *Pattern Matching* angewandt wurden, um die Suche nach häufigen Sequenzen effizienter zu gestalten [Wang u. a. 1994]. Da die Sequenzdatenbank in diesem Fall jedoch lediglich aus einer einzelnen, großen Zeichenkette besteht, ist die Übertragung der vorgeschlagenen Optimierungen auf andere Anwendungen nur eingeschränkt möglich.

### 2.2.3 Cluster-Analyse

Wie bereits erwähnt, besteht die grundlegende Zielstellung der Cluster-Analyse in der Gruppierung der vorhandenen Datenpunkte. Die resultierenden Gruppen (Cluster) sollten dabei möglichst homogen sein, d. h., Objekte innerhalb der gleichen Gruppe sollten sich möglichst ähnlich sein, während Objekte unterschiedlicher Gruppen möglichst wenig Ähnlichkeit untereinander aufweisen sollten. Um jedoch die Ähnlichkeit bzw. Unähnlichkeit zwischen zwei Objekten bestimmen zu können, muß zunächst ein funktionaler Zusammenhang zwischen den Ausprägungen der Eigenschaften der Objekte und dem daraus resultierenden Abstand zwischen den Objekten hergestellt werden. Eine solche Funktion wird im allgemeinen als Distanzfunktion bezeichnet und muß verschiedenen Kriterien genügen.

Seien $o_i$ und $o_j$ zwei Datenpunkte bzw. Objekte des betrachteten Objektraums $O$, dann müssen für eine Distanzfunktion *dist* die folgenden Bedingungen erfüllt sein [Ester und Sander 2000]:

1. $dist(o_i, o_j) = d \quad d \in \mathbf{R}, d \geq 0$
2. $dist(o_i, o_j) = 0 \quad$ genau dann, wenn $o_i = o_j$

[8] Das in [Agrawal und Srikant 1995] vorgestellte Modell legt fest, daß häufige Sequenzen maximal sind, d. h. nicht in einer anderen Sequenz enthalten sein dürfen. Deshalb können in diesem Fall strenggenommen keine derartigen Regeln abgeleitet werden.

3. $dist(o_i, o_j) = dist(o_j, o_i)$

Gilt zusätzlich auch noch die Beziehung $dist(o_i, o_j) + dist(o_j, o_k) \geq dist(o_i, o_k)$, $o_i, o_j, o_k \in O$, handelt es sich darüber hinaus um eine *Metrik*. Je größer die Distanz zwischen zwei Datenpunkten ist, desto unähnlicher sind sich die beiden Objekte. Wird statt dessen eine Ähnlichkeitsfunktion *sim* verwendet, erhöht sich ihr Wert mit zunehmender Ähnlichkeit der betrachteten Objekte. Die Wahl der *richtigen* Distanzfunktion stellt dabei immer eine besondere Herausforderung dar, da diese nicht nur den speziellen Anforderungen genügen muß, die sich aus der Art der zu analysierenden Daten ergeben, sondern auch zu einer für den Nutzer intuitiven und leicht verständlichen Gruppierung der Datenpunkte führen sollte.

Obwohl aufgrund der Vielzahl von Anwendungsmöglichkeiten eine große Anzahl verschiedener Verfahren für die Cluster-Analyse entwickelt wurde, lassen sich die meisten Algorithmen zwei grundlegend unterschiedlichen Gruppen von Methoden zuordnen. Partitionierende Verfahren erzeugen Cluster derart, daß in jedem Cluster mindestens ein Objekt enthalten ist und jedes Objekt zu genau einem Cluster gehört. Hierarchische Verfahren gruppieren Objekte dagegen auf unterschiedlichen Ebenen, so daß der gleiche Datenpunkt zu unterschiedlichen Gruppen gehören kann, je nachdem, welche Ebene der Hierarchie gerade betrachtet wird. Eine solche Hierarchie wird häufig als Baumstruktur dargestellt, wobei jeder Knoten einem Cluster entspricht. Die Wurzel des Baumes stellt dabei einen Cluster dar, der alle Datenpunkte $o_i \in O$ zusammenfaßt, während jedes Blatt einen Cluster repräsentiert, der aus einem einzelnen Objekt besteht. Die Distanz zwischen den Datenpunkten, die in einem Knoten zusammengefaßt sind, nimmt dabei von der Wurzel des Baumes zu den Blättern ab. Agglomerierende Verfahren konstruieren den Baum in den Blättern beginnend, d. h., jeder Datenpunkt bildet zunächst einen eigenen Cluster, welche schrittweise verschmolzen werden, bis alle Objekte im gleichen Cluster sind. Divergierende Verfahren gehen dagegen den entgegengesetzten Weg – sie starten mit einem einzelnen Cluster, den sie so lange teilen, bis jeder Datenpunkt einen separaten Cluster bildet.

Die meisten partitionierenden Verfahren arbeiten iterativ optimierend, d. h., sie erzeugen ein initiales Clustering, um dieses dann gemäß den vom Nutzer vorgegebenen Parametern zu optimieren (vgl. Abschnitt 2.4). Eines der bekanntesten Verfahren ist der Algorithmus *k-means*, bei dem der Nutzer lediglich die gewünschte Cluster-Anzahl $k$ vorzugeben braucht. Das initiale Clustering wird erstellt, indem zufällig $k$ Datenpunkte ausgewählt werden,

die dann jeweils einen Cluster bilden.[9] Anschließend werden die verbleibenden Datenpunkte jenem Cluster zugeordnet, für den der Wert der Funktion *dist* minimal ist. Im zweiten Schritt werden die *Centroide* der Cluster bestimmt und die Datenpunkte jeweils dem am nächsten gelegenen Centroid zugeordnet. Dieser zweite Schritt wird so lange wiederholt, bis sich die Grenzen der Cluster nicht mehr verschieben.

Das Optimalitätskriterium ist hierbei die Streuung der Datenpunkte, d. h., es wird der Abstand der Datenpunkte vom Centroid minimiert. Andere Verfahren verwenden anstelle des berechneten Mittelpunkts ein repräsentatives Objekt, welches als *Medoid* bezeichnet wird. Es gibt jedoch eine Reihe weiterer Kriterien, um ein optimales Clustering zu bestimmen. So werden bei *dichtebasierten* Verfahren untere Grenzen für die Anzahl von Objekten innerhalb eines festgelegten Radius angegeben, wodurch Cluster beliebiger Form gefunden werden können, deren Anzahl darüber hinaus im vorhinein auch nicht bekannt sein muß. Bei probabilistischen Verfahren werden dagegen die Wahrscheinlichkeiten bestimmt, mit denen ein gegebenes Objekt zu einem bestimmten Cluster gehört. Optimal ist in diesem Fall jenes Clustering, das die erwartete Wahrscheinlichkeit für die Cluster-Zugehörigkeit maximiert, d. h. für die gegebenen Daten am *plausibelsten* ist. Andere Algorithmen wurden speziell für die Verarbeitung von raumbezogenen Daten oder Daten mit kategorischen Attributen entwickelt.

### 2.2.4 Andere Mining-Techniken

Natürlich gibt es eine große Zahl weiterer Paradigmen, die sich jedoch häufig weniger in ihren Zielstellungen unterscheiden. Vielmehr stellen sie zumeist nur unterschiedliche Methoden dar, um vergleichbare Ziele zu erreichen, oder werden gemeinsam genutzt. Darüber hinaus finden diese Methoden oft auch in völlig anderen Gebieten Anwendung bzw. sind ursprünglich für andere Zwecke entwickelt worden. Auch wenn diese Zusammenstellung keinen Anspruch auf Vollständigkeit erhebt, sollen an dieser Stelle noch kurz die bereits erwähnte Klassifikation, die Generalisierung, das *Graph-Mining*, Neuronale Netze und Genetische Algorithmen beschrieben werden.

Bei der Klassifikation wird das bereits vorhandene Wissen in Form der Klassenzugehörigkeit eines Teils der Population genutzt, um eine Funkti-

[9] Da die Güte des resultierenden Clusterings sehr stark von der initialen Auswahl der $k$ Datenpunkte abhängt, wurden viele Verfahren zur Optimierung dieses Schrittes entwickelt.

on zur Ermittlung der Klassenzugehörigkeit unbekannter Instanzen zu lernen. Wichtige Vertreter sind z. B. *Bayes*-Klassifikatoren, die bedingte Wahrscheinlichkeiten für die Attributwerte der Klassen berechnen, um Aussagen über die Klassenzugehörigkeit der unbekannten Instanzen zu machen; das *Instance-based Learning* oder auch *Memory-based Reasoning*, das unbekannte Instanzen jener Klasse zuordnet, zu deren Vertretern die größte Ähnlichkeit besteht (*Nearest-Neighbor Classification*); Entscheidungsbäume, die aus den Trainingsdaten konstruiert werden und in jedem Knoten einen Test hinsichtlich des Wertes eines Attributs durchführen, um in den Blättern eine Klassenzugehörigkeit basierend auf den bisherigen Testergebnissen zu ermitteln.

Die Generalisierung dient in erster Linie dem besseren Verständnis der Daten, indem z. B. eine Reduktion der vorhandenen Dimensionen vorgenommen wird oder Aggregate berechnet werden. Viele dieser Operationen finden beim *Online Analytical Processing* (OLAP) Anwendung. Graphentheoretische Modelle und Methoden dienen als Grundlage für das Graph-Mining, bei dem es vor allem um die Analyse der *Beziehungen* zwischen Objekten geht. Dabei bilden die betrachteten Objekte, z. B. Kunden oder Produkte, die Knoten eines Graphen und die Beziehungen zwischen diesen Objekten die Kanten. Wie bei der Generalisierung besteht das Ziel häufig jedoch lediglich in einem besseren Verständnis der Daten, wofür nur eine Visualisierung der Beziehungen vorgenommen wird.

Die Funktionsweise Neuronaler Netze basiert auf dem Prinzip biologischer Gehirne. Ein solches Netz besteht aus Knoten, die den Neuronen eines Gehirns entsprechen und in Schichten angeordnet sind. Abgesehen von den Ein- und Ausgabeknoten, erzeugt jeder Knoten gemäß einer bestimmten Funktion aus seiner Eingabe eine Ausgabe, die er an einen oder mehrere Knoten, mit denen er verbunden ist, weiterleitet. Zusätzlich können die inneren Knoten auch mit unterschiedlichen Gewichten versehen werden. Durch Veränderung der Knotenzahl, ihrer Gewichte und/oder der Verbindungen zwischen Knoten kann dann die Ausgabe des Neuronalen Netzes bei gegebener Eingabe verändert werden. Neuronale Netze werden häufig zu Vorhersagezwecken eingesetzt, insbesondere auch zur Klassifikation, bei der die Klassenzugehörigkeit das Vorhersageziel darstellt. In diesem Fall wird das Netz so lange modifiziert, bis es die Klasse der Objekte in den Trainingsdaten korrekt bestimmen kann. Nach einer Test- und Optimierungsphase wird das Netz dann zur Bestimmung der Klasse unbekannter Instanzen verwendet.

Genetische Algorithmen bedienen sich ebenfalls eines biologischen Prin-

zips, dem der natürlichen Evolution, und werden u.a. zur Auswahl geeigneter Attribute für die Klassifikation (*Feature Selection*) oder auch zum Training Neuronaler Netze verwendet. Die Daten werden in diesem Fall als Menge binärer Attribut-Vektoren dargestellt, die als Gene bezeichnet und mit Hilfe der drei Operationen *Auslese*, *Kreuzung* und *Mutation* verändert werden. Ziel des iterativ optimierenden Verfahrens ist die Maximierung bzw. Minimierung einer festgelegten Zielfunktion. In jeder Iteration wird im Rahmen der Auslese eine bestimmte Anzahl jener Gene ausgewählt, die den Zielfunktionswert der Population verbessern. Im Rahmen der Kreuzung werden diese dann kombiniert, z. B. mit Hilfe logischer Operationen, bzw. im Rahmen der Mutation zufällig verändert. Anschließend wird der Wert der Zielfunktion erneut berechnet. Kann dieser nicht weiter verbessert werden, bricht das Verfahren ab.

Die Auswahl der vorgestellten Mining-Verfahren ist natürlich nicht vollständig. Wie am Ende des letzten Kapitels bereits betont, war es nicht beabsichtigt, einen *umfassenden* Überblick über die Methoden und Techniken des Data Mining zu geben, zumal eine Darstellung der spezifischen Vor- und Nachteile der verschiedenen Methoden und, in Abhängigkeit davon, ihrer Eignung für bestimmte Analysezwecke wohl den Rahmen dieses Kapitels sprengen würde. Statt dessen sollte jedoch deutlich geworden sein, daß es eine Vielzahl von Methoden gibt, die zum einen dem besseren Verständnis der Daten dienen, zum anderen jedoch auch *neues*, explizites Wissen aus den vorhandenen Daten generieren können. Da sich diese Arbeit mit der Evolution des entdeckten Wissens beschäftigt, ist die Gestalt der zu erwartenden Ergebnisse natürlich von besonderem Interesse. Während in diesem Abschnitt die prinzipielle Arbeitsweise grundlegender Techniken der Wissensentdeckung beschrieben wurde, soll in den verbleibenden Abschnitten dieses Kapitels auf die Fragen eingegangen werden, in welcher Form das generierte Wissen repräsentiert wird und wie die Validität bzw. Evidenz des Wissens festgestellt werden kann.

## 2.3 Mining-Ergebnisse

In Abhängigkeit davon, welches Paradigma Verwendung findet, kann sich die Form der Mining-Ergebnisse beträchtlich unterscheiden. Hinzu kommt, daß die Darstellung der Ergebnisse selbst bei Verwendung des gleichen Paradigmas unterschiedlich erfolgen kann, je nachdem, welches Verfahren zur

Anwendung kommt.

Für die Darstellung der Ergebnisse einer Warenkorbanalyse werden im allgemeinen die bereits genannten Assoziationsregeln verwendet. Diese Regeln stellen Implikationen der Form $X \Rightarrow Y$ dar, d. h. Aussagen der Form „*Wenn* $X$*, dann* $Y$“, wobei $X$ und $Y$ häufige Itemsets sind (vgl. Abschnitt 2.2.1). Ein Item ist in diesem Fall ein Literal, d. h. eine Zeichenkette, die als Bezeichner für das jeweils betrachtete Objekt dient. Es können jedoch auch gerichtete Graphen zur Visualisierung von Assoziationsregeln verwendet werden, wobei die Produkte bzw. Items die Knoten des Graphen bilden und die Implikationen die Kanten.

Für die Ergebnisse der Sequenzanalyse kommen in Abhängigkeit vom zugrunde liegenden Modell unterschiedliche Darstellungsweisen in Betracht. Im einfachsten Fall ist eine Sequenz $\mathsf{s}$ eine geordnete Folge von Zeichen, z. B. Buchstaben eines Alphabets, wobei $\mathsf{s}$ eine Subsequenz der Sequenzdatenbank $S$ ist, d. h. $\mathsf{s} \subseteq S$. Findet das in [Agrawal und Srikant 1995] vorgestellte Modell Anwendung, wird grundsätzlich die gleiche Darstellungsweise verwendet, es handelt sich jedoch in diesem Fall um eine Folge von Transaktionen, deren relative Ordnung durch die zeitliche Reihenfolge der Transaktionen gegeben ist. Konkret sind dann eine Folge von Kundentransaktionen gemeint, die jeweils aus nichtleeren Itemsets bestehen und in aufsteigender Reihenfolge des Transaktionszeitpunktes sortiert sind. Bei dem in [Mannila u. a. 1997] beschriebenen Modell werden Ereignisse aus einem gegebenen Ereignisraum betrachtet, die eine absolute Ordnung aufweisen. Eine Sequenz ist in diesem Fall eine Folge von Ereignissen, wobei jedem Ereignis ein eindeutiger Zeitstempel zugeordnet ist. Für die Darstellung wird dann das im Abschnitt 2.2.2 vorgestellte Tripel $(s, T_S, T_E)$ genutzt, wobei die eigentliche Sequenz in $s$ abgelegt ist. Zusätzlich werden bei Verwendung dieses Modells auch Assoziationsregeln der Form $\beta \Rightarrow \gamma$ abgeleitet, wobei die linke und die rechte Seite der Regel häufige Sequenzen in $S$ sind und $\beta \preceq \gamma$ gelten muß.

Es wird deutlich, daß es sich bei beiden Regeltypen um stark strukturierte Objekte handeln kann, für die eine Repräsentation als einfache Zeichenketten nicht ausreichend ist. Hinzu kommt, daß für eine Regel eine Anzahl statistischer Eigenschaften erfaßt werden, die bei der Darstellung der Ergebnisse ebenfalls zu berücksichtigen sind (vgl. Abschnitt 2.4). Gleiches gilt für die Ergebnisse einer Cluster-Analyse, für die in Abhängigkeit von der jeweils verwendeten Methode unterschiedliche Repräsentationen genutzt werden. Im einfachsten Fall wird ein Cluster durch die Objekte repräsentiert, die in ihm enthalten sind. Ist die Anzahl der Dimensionen sehr klein

($\leq 3$), finden dafür oft graphische Darstellungen Anwendung. Bei höherdimensionalen Daten lassen sich häufig noch Projektionsverfahren nutzen, die aber mit einem kalkulierten Informationsverlust verbunden sind. Gerade bei hierarchischen Verfahren kann diese Art der Darstellung jedoch die Verständlichkeit der erzielten Ergebnisse deutlich erhöhen. Im Falle partitionierender Verfahren wird ein Cluster üblicherweise durch seinen Centroid oder, falls das Clustering auf repräsentativen Objekten beruht, durch seinen Medoid identifiziert. Dieser Punkt wird durch einen Vektor dargestellt, dessen Länge sich aus der Anzahl der Dimensionen der Datenpunkte ergibt. Alle anderen Eigenschaften eines Clusters, z. B. seine statistischen Merkmale oder die Zuordnung der Datenpunkte zu einem Cluster, können bei Bedarf aus den Basisdaten ermittelt werden. Alternativ kann die Zugehörigkeit eines Objekts zu einem Cluster auch als zusätzliches Attribut (*Label*) in den Daten erfaßt werden, was insbesondere bei dichtebasierten Verfahren üblich ist, da es hier im allgemeinen keine zentralen bzw. repräsentativen Objekte gibt. In einem solchen Fall kann die Darstellung des Clusterings auch durch die Clustergrenzen erfolgen: Ist $n$ die Anzahl der Dimensionen des Datenraums, werden die Grenzen der Cluster durch Hyperebenen der Dimension $n - 1$ abgebildet. Bei probabilistischen Verfahren wie dem EM-Algorithmus (*Expectation Maximization*) wird ein Clustering dagegen in Form einer Wahrscheinlichkeitsverteilung dargestellt, d. h., für die Dimension eines Clusters werden die Parameter $\mu$ und $\sigma$ der Dichtefunktion angegeben.

Klassifikationswissen läßt sich beispielsweise in Form von Regeln darstellen, die Assoziationsregeln ähneln. Hingegen stellen Klassifikationsregeln jedoch keine Beziehung zwischen den Objekten des Datenraums her, sondern repräsentieren Tests auf ihren Attributwerten. Die linke Seite der Regel stellt dabei den Test dar, und die rechte Seite nimmt – in Abhängigkeit vom Testergebnis – eine Zuordnung des betrachteten Objekts zu einer der Klassen vor. Eine alternative Darstellungsmöglichkeit bilden Entscheidungsbäume. Hier repräsentiert ein Knoten eine Entscheidung bezüglich eines Attributwertes, die im Fall eines inneren Knotens weitere Entscheidungen nach sich zieht oder, in den Blättern des Baumes, die Klassenzugehörigkeit festlegt. Durch die logische Verknüpfung der Entscheidungen in den Knoten läßt sich ein Entscheidungsbaum auch in eine Menge von Klassifikationsregeln transformieren.

Die anderen im Abschnitt 2.2.4 beschriebenen Techniken generieren zumeist kein explizites Wissen aus den analysierten Daten. Obwohl sich z. B. Neuronale Netze gut zur Klassifikation eignen und oft genauere Ergebnisse

liefern als Entscheidungsbäume, läßt sich aus einem solchen Netz kein Klassifikationswissen ableiten: Natürlich kann die Topologie des trainierten Netzes als Graph abgebildet werden, aber es liefert z. B. keine Klassenbeschreibungen, die einem besseren Verständnis der Klassen dienen könnten.

## 2.4 Qualitätsmaße

Obwohl die Anzahl der gefundenen Muster im Vergleich zur Größe des analysierten Datensatzes im allgemeinen um Größenordnungen kleiner ist, generieren viele Algorithmen immer noch eine für den Anwender kaum überschaubare Menge von Ergebnissen. Aus diesem Grund wurden verschiedene Maßzahlen entwickelt, die dem Nutzer die Bewertung der *Wichtigkeit* bzw. Güte der gefundenen Zusammenhänge und Modelle gestatten.

Bei der Bewertung von Assoziationsregeln haben sich die Maße Support und Konfidenz durchgesetzt, für welche im allgemeinen untere Grenzen als Analyse-Parameter angegeben werden. Eigentlich geben beide Maße den gleichen Sachverhalt wieder – den Anteil der Instanzen, für den die Regel eine korrekte Vorhersage macht. Während sich der Support dabei jedoch auf die Menge *aller* Instanzen bezieht, kommen zur Berechnung der Konfidenz nur jene Instanzen in Betracht, die von der Assoziation überhaupt betroffen sind. Der Support $s$ entspricht der relativen Häufigkeit des jeweiligen Itemsets, d. h. dem Anteil der Transaktionen, die das betrachtete Itemset enthalten. Bezeichnet $|X|$ die Anzahl der Transaktionen, die das Itemset $X$ enthalten, und $|D|$ die Anzahl der Transaktionen in der Datenbank $D$, dann ergibt sich der Support des Itemsets $X$ aus

$$s(X) = \frac{|X|}{|D|}, \qquad s \in \mathbf{R}, 0 \leq s \leq 1.$$

Der Support einer Assoziationsregel $X \Rightarrow Y$ ergibt sich analog aus

$$s(X \Rightarrow Y) = \frac{|X \cup Y|}{|D|},$$

wobei $|X \cup Y|$ die Anzahl der Transaktionen bezeichnet, welche die Itemsets $X$ *und* $Y$ enthalten.

Die Konfidenz $c$ einer Assoziationsregel ist ein Maß für die *Stärke* der Assoziation. Sie bezeichnet die bedingte Wahrscheinlichkeit für das Ergebnis

der Implikation bei gegebener Voraussetzung und ergibt sich als Anteil jener Transaktionen, die beide Seiten der Regel enthalten, an den Transaktionen, welche die linke Seite der Regel unterstützen, d. h.

$$c(X \Rightarrow Y) = \frac{|X \cup Y|}{|X|} \quad \text{bzw.}$$

$$= \frac{s(X \Rightarrow Y)}{s(X)}, \qquad c \in \mathbf{R}, 0 \leq c \leq 1.$$

Häufig wird für eine Assoziationsregel auch die *erwartete* Konfidenz angegeben, die sich als Quotient aus der Anzahl der Transaktionen, welche die rechte Seite der Assoziation enthalten, und der Gesamtzahl der Transaktionen in $D$ ergibt:

$$ec(X \Rightarrow Y) = \frac{|Y|}{|D|}, \qquad ec \in \mathbf{R}, 0 \leq ec \leq 1.$$

Ein weiteres Maß, das die Einschätzung der Qualität einer Assoziationsregel erlaubt, ist der sogenannte *Lift*, häufig auch als *Improvement* bezeichnet. Er ergibt sich als Quotient aus der Konfidenz und der erwarteten Konfidenz:

$$l(X \Rightarrow Y) = \frac{c(X \Rightarrow Y)}{ec(X \Rightarrow Y)} \quad \text{bzw.}$$

$$= \frac{s(X \Rightarrow Y)}{s(X) \cdot s(Y)}, \qquad l \in \mathbf{R}, l \geq 0,$$

wobei Werte $l < 1$ bedeuten, daß die Regel zu Vorhersagezwecken ungeeignet ist.[10] Ein wesentlicher Nachteil dieser Maßzahl besteht darin, daß der Wertebereich keine obere Grenze aufweist. Der *Certainty Factor* behebt dieses Problem und macht darüber hinaus – ähnlich dem Korrelationskoeffizienten – eine Aussage, die das Vorzeichen des Zusammenhangs zwischen der linken und der rechten Seite einer Assoziation betrifft [Shortliffe und Buchanan 1975]:

$$cf(X \Rightarrow Y) = \begin{cases} \dfrac{(c(X \Rightarrow Y) - s(Y))}{(1 - s(Y))} & \text{für } c(X \Rightarrow Y) > s(Y), \\ (c(X \Rightarrow Y) - s(Y)) \cdot s(Y) & \text{sonst, } cf \in \mathbf{R}, -1 \leq cf \leq 1. \end{cases}$$

[10] Viele Mining-Algorithmen verwerfen deshalb Regeln, für die $l < 1$ gilt.

Natürlich gibt es noch viele andere Maßzahlen, die Aussagen über die Qualität der gefundenen Muster erlauben [Tan und Kumar 2000]. Bei ihrer Auswahl ist jedoch in jedem Fall der Anwendungskontext zu berücksichtigen, da er direkte Auswirkungen auf die Eignung eines Maßes haben kann [Tan u. a. 2002].

Für die Ergebnisse der Sequenzanalyse lassen sich prinzipiell die gleichen Maßzahlen verwenden, es ist jedoch zu beachten, daß der Support einer häufigen Sequenz auf andere Art bestimmt wird. Besteht eine Sequenz aus einer einfachen Zeichenkette, wird die absolute Häufigkeit der betrachteten Zeichenkette, d. h. die Anzahl ihres Auftretens, ins Verhältnis zur Gesamtlänge der Sequenzdatenbank gesetzt. Beim in [Agrawal und Srikant 1995] beschriebenen Modell wird dagegen die Anzahl der Kunden, die die entsprechende Transaktionshistorie aufweisen, ins Verhältnis zur Gesamtzahl der in der Datenbank erfaßten unterschiedlichen Kunden gesetzt. Bei dem in [Mannila u. a. 1997] beschriebenen Modell schließlich wird der Support einer Episode durch den Quotienten aus der Anzahl der Zeitfenster, welche die jeweilige Episode enthalten, und der Gesamtzahl der Zeitfenster bestimmt. Da in diesem Fall die Gesamtzahl der Zeitfenster von der gewählten Breite abhängt, kann dieselbe Episode unterschiedliche Support-Werte aufweisen.[11]

Für die Ergebnisse der Cluster-Analyse ist die Güte im allgemeinen nur im Rahmen des Anwendungskontextes zu bestimmen. Generell läßt sich die Qualität der gefundenen Cluster jedoch in Abhängigkeit von der gewählten Distanzfunktion ermitteln. Verfahren der Cluster-Analyse, die auf der Auswahl zentraler Punkte beruhen, bestimmen die *Kompaktheit* eines Clusters $C$ als Summe der quadrierten euklidischen Distanzen zwischen den Objekten des Clusters und seinem Centroid $\mu_C$:

$$TD^2(C) = \sum_{o \in C} dist\,(o, \mu_C)^2, \qquad TD^2 \in \mathbf{R},\, TD^2 \geq 0.$$

Die Kompaktheit für das gesamte Clustering ergibt sich aus der Summe der Werte für alle Cluster:

$$TD^2 = \sum_{i=1}^{k} TD^2\,(C_i)$$

Für Verfahren, die auf der Auswahl repräsentativer Objekte basieren, wird ein vergleichbares Maß berechnet, wobei üblicherweise jedoch nur der einfache

[11] Für eine gegebene Ereignissequenz $\mathtt{s} = (s, T_S, T_E)$ und ein Zeitfenster der Breite $w$ ergibt sich die Anzahl der Zeitfenster aus $T_E - T_S + w - 1$ (vgl. [Mannila u. a. 1997]).

Abstand zwischen den Objekten und dem Medoid $m_C$ einbezogen wird:

$$\begin{aligned} TD(C) &= \sum_{o \in C} dist\,(o, m_C) \quad \text{bzw.} \\ TD &= \sum_{i=1}^{k} TD\,(C_i)\,, \qquad TD \in \mathbf{R},\, TD \geq 0. \end{aligned}$$

Darüber hinaus kann die durchschnittliche Distanz innerhalb der Cluster auch mit der durchnittlichen Distanz zwischen den Clustern verglichen werden.

Für hierarchische Verfahren ist die Güte des Clusterings schwerer zu bestimmen, da die Streuung in der Wurzel des Baumes immer maximal und in den Blättern minimal ist. Eine Möglichkeit besteht jedoch darin, die Werte der Distanzfunktion für aufeinanderfolgende Knoten des Baumes zu vergleichen. Große Differenzen deuten dabei immer auf *starke* Cluster hin. Da das Ergebnis dichtebasierter Verfahren nicht vom initialen Clustering abhängt, wird für gegebene Parameter immer das gleiche Clustering erzeugt, und insofern ist eine Veränderung der Güte des Clusterings nur durch eine Veränderung der Parameter zu erreichen. Bei probabilistischen Verfahren ergibt sich die Qualität des Clusterings aus dem Logarithmus der Wahrscheinlichkeit für die Daten unter der jeweils angenommenen Wahrscheinlichkeitsverteilung. Je größer dieser Wert, desto besser ist auch das gefundene Clustering.

Für die Klassifikation wird üblicherweise bestimmt, mit welcher Rate der Klassifikator die Klassenzugehörigkeit korrekt ermittelt. Sei $O_{pos}$ die Menge der korrekt klassifizierten Objekte und $D_{test}$ der Testdatensatz, dann ergibt sich die Klassifikationsgenauigkeit (engl. *accuracy*) aus

$$a = \frac{|O_{pos}|}{|D_{test}|}, \qquad a \in \mathbf{R}, 0 \leq a \leq 1.$$

Der Klassifikationsfehler ergibt sich analog aus

$$\begin{aligned} e &= \frac{|O_{neg}|}{|D_{test}|} \quad \text{bzw.} \\ &= 1 - a, \end{aligned}$$

wobei $O_{neg}$ die Menge der falsch klassifizierten Objekte bezeichnet. Hierbei handelt es sich um den sogenannten *wahren* Klassifikationsfehler (engl. *true*

*error*), im Gegensatz zum *beobachteten* Klassifikationsfehler (engl. *apparent error*), der sich aus

$$e_a = \frac{|O_{neg}|}{|D_{training}|}$$

ergibt, wobei $D_{training}$ die Trainingsdaten bezeichnen.[12]

Für die Bewertung von Mining-Ergebnissen gibt es noch sehr viele andere Kriterien, deren Eignung sich im Hinblick auf den Anwendungskontext und die verwendeten Methoden unterscheidet (vgl. z. B. [Freitas 1998; Gago und Bento 1998]). Es ist jedoch offensichtlich, daß die Aussagekraft der gefundenen Modelle und Muster in entscheidendem Maße von den oben beschriebenen Kriterien, die im folgenden als *statistische Eigenschaften* eines Musters bezeichnet werden, abhängt. Ungeachtet der Frage, welche Maße tatsächlich verwendet werden, sollten das Muster und seine statistischen Eigenschaften deshalb als Einheit gesehen werden.

[12] Eigentlich ist die Bezeichnung des wahren Klassifikationsfehlers irreführend, da er nur für die Testdaten bestimmt werden kann. Für die Grundgesamtheit wird er im Normalfall geschätzt.

# Kapitel 3

# Literaturüberblick

Konventionelle Algorithmen des Data Mining gehen bei der Analyse von statischen Datensätzen aus. Dabei wird die Tatsache vernachlässigt, daß der überwiegende Teil der gesammelten Daten dynamischer Natur ist. Abgesehen von der Frage, in welchem Anwendungsbereich die Daten gesammelt werden, ist der Anwendungsexperte im Normalfall jedoch mit einem *evolvierenden* Datensatz konfrontiert: Transaktionsdaten, die die Verkäufe in einem Supermarkt oder die Zugriffe auf einen Server protokollieren, werden fortlaufend erweitert; Zeitreihen, die die Entwicklung von Aktienkursen oder Witterungsbedingungen erfassen, werden beständig fortgeschrieben. Während es im Fall von operationalen Datenbanken auch zu Löschungen und, wie z. B. bei der Verwaltung von Nutzerprofilen, zu teilweise massiven Aktualisierungen im erfaßten Datenbestand kommen kann, handelt es sich beim weitaus größeren Teil dieser Änderungen – notwendige Korrekturen bei historischen Daten ausgenommen – ausschließlich um neue Einträge. Hinzu kommt, daß die jeweils beobachtete Population sich ändernden Rahmenbedingungen ausgesetzt ist, die nicht in den gesammelten Daten erfaßt sind: Konsumenten reagieren auf Marketing-Kampagnen oder geänderte rechtliche Regularien, der Datenbestand eines WWW-Servers kann umstrukturiert oder geändert werden. Solche Änderungen üben jedoch sehr wahrscheinlich einen Einfluß auf die Ergebnisse aus, die eine Analyse der gesammelten Daten liefert. Hierbei sind zwei unterschiedliche Aspekte zu beachten: Zum einen ist es für den Anwendungsexperten von großer Wichtigkeit, wie sich die Änderungen in den Daten auf die Analyse-Ergebnisse auswirken, d. h., führen die Änderungen zum Entstehen neuer Muster, oder können bislang existierende Muster nicht mehr gefunden werden. Zum anderen ist es für den Anwender wichtig

zu wissen, zu welchem Zeitpunkt sich bestimmte Änderungen in den Analyse-Ergebnissen manifestieren und welche davon für ihn interessant sind.

In dieser Arbeit wird Data Mining als iterativer Prozeß verstanden: Die gesammelten Daten werden periodisch im Rahmen sogenannter Mining-Sessions untersucht. Jede Mining-Session liefert dabei eine Menge von Mustern zurück, von denen ein Teil aus früheren Sitzungen bekannt ist, andere jedoch neu sind und, möglicherweise, unerwartet zutage treten. Für die vorliegende Arbeit sind in diesem Zusammenhang verschiedene Problemstellungen relevant. Im folgenden Abschnitt soll zunächst auf solche Publikationen eingegangen werden, die sich dem ersten Aspekt des Einflusses von Änderungen in den Daten auf die Ergebnisse der Analyse widmen. Der Schwerpunkt dieser Arbeiten liegt in der Beantwortung der Frage, *wie* die Muster aktualisiert werden sollen. Im Abschnitt 3.2 wird es um die Erkennung der Art von Änderungen gehen, denen die Muster ausgesetzt sind, wobei auch Beiträge diskutiert werden, die erörtern, *wann* die entdeckten Muster von Änderungen betroffen sind.

Aufgrund der Vielzahl von Regeln, die z. B. im Rahmen einer Warenkorbanalyse gefunden werden können, stellt sich zusätzlich die Frage, *welche Regeln* für den Anwender besonders interessant sind. Häufig ist es aus praktischen Erwägungen unmöglich, die Änderungen aller Muster zu überwachen. Aus diesem Grund ist es notwendig, eine geeignete Vorauswahl zu treffen. Eng mit dieser Problematik verknüpft ist die Frage, *welche Änderungen* dem Nutzer gemeldet werden sollen: Läßt man den relativ unwahrscheinlichen Fall außer acht, daß sich ein Muster im Zeitverlauf überhaupt nicht ändert, steigt die Menge der beobachtbaren Musteränderungen mit jeder Mining-Session an. Im Abschnitt 3.3 wird auf Arbeiten eingegangen, die sich mit dieser Problemstellung befassen. Im Abschnitt 3.4 werden anschließend Arbeiten besprochen, die sich mit dem Phänomen der Konzeptdrift, der eigentlichen Ursache von Musteränderungen, auseinandersetzen.

Wie im Abschnitt 1.1 beschrieben wurde, liegt das Ziel der Analysen unter anderem darin, von den beobachteten Musteränderungen auf interessante Änderungen in den zugrunde liegenden Daten zu schließen. Als Resultat der Analyse liegen dem Anwender neben den interessanten Änderungen auch die Historien der erfaßten statistischen Eigenschaften all jener Muster vor, die er, entweder nach konventionellen Kriterien oder aufgrund ihrer Änderungen im Zeitverlauf, für relevant hält. Diese Daten stehen natürlich auch für zusätzliche Analysen zur Verfügung. Neben den klassischen Verfahren der Zeitreihenanalyse sind dabei auch die im Abschnitt 3.5 betrachteten Arbei-

ten zur Suche nach Mustern in Zeitreihen relevant.

## 3.1 Inkrementelle Mining-Techniken

Das Ziel inkrementeller Mining-Algorithmen liegt in der Aktualisierung der erhaltenen Mining-Ergebnisse, wobei der Schwerpunkt der vorgestellten Methoden häufig in der Effizienz, mit der die Aktualisierung durchgeführt wird, zu sehen ist. Im allgemeinen wird eine Datenbank $D$ betrachtet, deren Analyse eine Menge von Mustern $P$ produziert hat und die im Laufe der Zeit einer Reihe von Änderungen $\delta$ ausgesetzt war. Im Rahmen der Analyse wird dann versucht, die Auswirkungen von $\delta$ auf $P$ zu ermitteln. Natürlich wäre es ineffizient, eine grundlegend neue Analyse auf der aktualisierten Datenbank $D + \delta$ vorzunehmen. Da üblicherweise davon ausgegangen wird, daß $|D| \gg |\delta|$, versuchen die verschiedenen Techniken, die Anzahl der Zugriffe auf $D$ so gering wie möglich zu halten und möglichst nur mit $P$ und $\delta$ zu arbeiten. Ein großer Teil der Algorithmen ist darüber hinaus in der Lage, die Aktualisierung auch dann durchzuführen, wenn $\delta$ nicht nur aus Einfügungen besteht, sondern auch Löschungen enthält.

Die meisten der im folgenden vorgestellten Arbeiten sind in [Baron und Spiliopoulou 2002] beschrieben, wo auch ein Bezugssystem entwickelt wurde, das einen Vergleich dieser Arbeiten ermöglicht.

### 3.1.1 Aktualisierung von Assoziationsregeln

Der weitaus größere Teil der Beiträge im Bereich der inkrementellen Mining-Verfahren befaßt sich mit der Aktualisierung von Assoziationsregeln. Die Familie der FUP-Algorithmen (Fast UPdate) verwendet Informationen aus zurückliegenden Mining-Sessions in Form von $P$, um die Bestimmung der Auswirkungen von $\delta$ effizienter zu gestalten [Cheung u. a. 1996a, b, 1997]. Die Vorgehensweise von FUP entspricht dabei im wesentlichen den bekannten Algorithmen *Apriori* [Agrawal und Srikant 1994] und DHP [Park u. a. 1995]: In $k$ Iterationen werden die häufigen Itemsets $L_k^{D+\delta}$ bestimmt, wobei jedoch die *alten* häufigen Itemsets $L_k^D$ und ihre Support-Werte wiederverwendet werden, um ausschließlich anhand des Inkrements $\delta$ zu entscheiden, welche alten Regeln dem vorgegebenen Support-Kriterium nicht mehr genügen bzw. welche neue Regeln hinzugekommen sind. Wie bereits erwähnt wurde, liegt das Ziel der meisten inkrementellen Methoden in einer möglichst effi-

zienten Aktualisierung der gefundenen Muster. FUP erreicht dieses Ziel im wesentlichen durch das frühzeitige Entfernen von Regel-Kandidaten, die in der aktualisierten Datenbank $D + \delta$ nicht häufig sein können. Aufgrund der Wiederverwendung von Informationen aus früheren Sitzungen geschieht dies ausschließlich unter Verwendung von $\delta$, das im allgemeinen sehr viel kleiner ist als $D$. Basierend auf FUP, werden zwei zusätzliche Algorithmen FUP* and MLUp vorgeschlagen, wovon ersterer eine verbesserte Version des ursprünglichen Algorithmus FUP darstellt und der zweite eine Variante für die Aktualisierung von Multi-Level Assoziationsregeln ist [Cheung u. a. 1996b]. In [Cheung u. a. 1997] schließlich wird ein Algorithmus $\text{FUP}_2$ vorgeschlagen, der eine Generalisierung von FUP darstellt und die Aktualisierung auch dann vornehmen kann, wenn $\delta$ aus Einfügungen *und* Löschungen besteht.

Der in [Ayan u. a. 1999] vorgeschlagene Algorithmus *UWEP* (Update With Early Pruning) verfolgt eine vergleichbare Strategie. Im Gegensatz zu den FUP-Algorithmen wird jedoch ein Zwischenspeicher verwendet, der eine Liste von Transaktionskennungen und den dazugehörigen Statistiken enthält. Auf diese Weise wird die Anzahl der notwendigen Lesezugriffe stark vermindert, was zu einer zusätzlichen Effizienzsteigerung führt.

In [Omiecinski und Savasere 1998] wird eine Variation des von Savasere u. a. [1995] entwickelten *Partition*-Algorithmus zur Assoziationsregelentdeckung vorgeschlagen. Hierbei bilden $D$ und $\delta$ die beiden Partitionen. Der Algorithmus arbeitet in zwei Phasen. Im Fall, daß $\delta$ ausschließlich aus Einfügungen besteht, werden in der ersten Phase alle häufigen Itemsets und die Gesamtzahl der Transaktionen in $\delta$ bestimmt. Wieder werden die Ergebnisse aus zurückliegenden Sitzungen verwendet, so daß $D$ nicht eingelesen werden muß. In der zweiten Phase wird anschließend überprüft, ob die Itemsets, die in den Partitionen häufig sind, dem vorgegebenen Support-Kriterium auch global genügen. Enthält $\delta$ auch Einfügungen neuer bzw. Aktualisierungen bestehender Datensätze, müssen in der ersten Phase zudem die Statistiken in $D$ aktualisiert werden. Auch bei diesem Algorithmus wird die Effizienz der Aktualisierung durch die reduzierte Anzahl an notwendigen Lesezugriffen in $D$ und $\delta$ erreicht.

Der in [Thomas u. a. 1997] vorgestellte Algorithmus basiert auf dem Konzept der sogenannten *Negative Border*, das zuerst in [Toivonen 1996] vorgestellt wurde. Es handelt sich dabei um die Menge der Kandidaten häufiger $k$-Itemsets, die dem Support-Kriterium nicht genügten (vgl. Abschnitt 2.2.1). Bezeichnet $C_k$ die Menge der $k$-Itemsets, deren Support zu überprüfen ist, und $L_k$ die Menge der häufigen $k$-Itemsets, dann ergibt sich die Negative

Border von $L_k$ aus $NB(L_k) = C_k - L_k$. Im ersten Schritt ermittelt der Algorithmus die häufigen Itemsets und den Support der Itemsets $L_k^D$ und $NB(L_k^D)$ in $\delta$. Ist ein Itemset dabei nicht häufig in $D + \delta$, wird es entfernt und die Negative Border neu bestimmt. Dann wird für jedes häufige Itemset $L_k^\delta$ überprüft, ob es in der aktualisierten Datenbank $D + \delta$ genügend Support aufweist, um von $NB(L_k^D)$ in $L_k^{D+\delta}$ zu wechseln. Ist die Vereinigung der Itemsets $L_k^{D+\delta}$ und $NB(L_k^{D+\delta})$ anschließend nicht gleich der Vereinigung der Itemsets $L_k^D$ und $NB(L_k^D)$ wird die gesamte Datenbank $D + \delta$ einmal eingelesen, um die neue Negative Border $NB(L_k^{D+\delta})$ zu bestimmen.[1] Anders als bei den bisher beschriebenen Verfahren werden hier zusätzliche Informationen in Form der Itemsets in $NB(L_k^D)$ zur Aktualisierung der Muster in $P$ genutzt. Dabei stehen der weiteren Reduzierung der Lesezugriffe durch die Verwendung der Negative Border aber die zusätzlichen Anforderungen, die sich aus ihrer Speicherung und Aktualisierung ergeben, gegenüber.

### 3.1.2 Aktualisierung häufiger Sequenzen

Die Anzahl der Arbeiten im Bereich der Aktualisierung von häufigen Sequenzen ist deutlich kleiner als für die Warenkorbanalyse. In den Arbeiten von Wang u. a. wird eine spezielle Indexstruktur vorgeschlagen, um eine Aktualisierung der häufigen Sequenzen vorzunehmen [Wang und Tan 1996; Wang 1997]. Im allgemeinen läßt sich das Finden häufiger Sequenzen wie bei *Apriori* in zwei Schritte zerlegen: Im ersten Schritt werden alle häufigen Sequenzen gemäß $\tau_s$ gesucht, im zweiten Schritt werden aus diesen Sequenzen unter Berücksichtigung von $\tau_c$ Regeln abgeleitet (vgl. Abschnitt 2.2.2). Die grundlegende Idee in diesem Fall ist jedoch, daß die Indexstruktur nicht nur den Inhalt der Sequenzdatenbank widerspiegelt, sondern darin gleichzeitig die Support-Werte der Subsequenzen erfaßt werden. Wird der vorgeschlagene Index, eine Variation eines Suffix-Baumes, *Dynamic Suffix Tree* genannt, für die Indexierung der Sequenzdatenbank verwendet, können häufige Sequenzen somit direkt aus der Baumrepräsentation der Datenbank gelesen werden. Im Falle von Änderungen der Datenbank erfolgt die Aktualisierung der häufigen Sequenzen dann automatisch im Zuge der Aktualisierung der Indexstruktur, für die ein effizienter Algorithmus vorgeschlagen wird. Die neuen häufigen Sequenzen können dem Baum anschließend wieder direkt entnommen werden.

[1] Ein vergleichbarer Ansatz wurde zeitgleich in [Feldman u. a. 1997] vorgeschlagen.

Andere vorgeschlagene Techniken nutzen, wie schon die verschiedenen Methoden zur Aktualisierung von Assoziationsregeln, die Ergebnisse aus früheren Mining-Sessions, um eine effiziente Aktualisierung der gefundenen Muster durchzuführen [Parthasarathy u. a. 1999; Masseglia u. a. 2000; Zhang u. a. 2002a]. In [Zhang u. a. 2002b] wurde darüber hinaus ein Vergleich bezüglich der Performanz der verschiedenen Methoden vorgenommen.

### 3.1.3 Aktualisierung von Cluster-Beschreibungen

In [Ester u. a. 1998] wird ein Algorithmus zur inkrementellen Aktualisierung der Ergebnisse einer Cluster-Analyse vorgeschlagen. Der Algorithmus *IncrementalDBSCAN* basiert auf seinem statischen Vorgänger DBSCAN, ein dichtebasiertes Verfahren für die Analyse metrischer Daten in Data-Warehouse-Umgebungen. Ziel der Methode ist die Identifizierung der Auswirkungen von Aktualisierungen im Warehouse und die Anpassung des ermittelten Clusterings. Die Idee dichtebasierter Verfahren besteht darin, daß für jedes Objekt eines Clusters in einem gegebenen Radius mindestens $n$ andere Objekte existieren müssen. Der Radius und die Anzahl $n$ sind dabei vom Nutzer vorzugeben (vgl. Abschnitt 2.2.3). Neben Kern- und Grenzobjekten werden dabei Datenpunkte, die keinem Cluster zugeordnet sind – sogenanntes Rauschen –, unterschieden. Änderungen im Warehouse können dazu führen, daß Kernobjekte in einen anderen Cluster migrieren, sich die Cluster-Grenzen verschieben und/oder sich die Dichteerreichbarkeit von Objekten verändert. Um solche Fälle zu erkennen, bedient sich *IncrementalDBSCAN* verschiedener Heuristiken. Aufgrund der Eigenschaften eines dichtebasierten Clusterings können nur solche Objekte von einer Einfügung oder Löschung betroffen sein, die sich innerhalb des vom Nutzer vorgegebenen Radius um den einzufügenden bzw. zu löschenden Datenpunkt befinden, wodurch sich die Aktualisierung sehr effizient gestalten läßt. Obwohl die vorgeschlagene Methode selbst eigentlich kein Clustering-Algorithmus ist, wird formal gezeigt, daß die Anwendung von *IncrementalDBSCAN* zum gleichen Ergebnis führt wie eine erneute Analyse der aktualisierten Datenbank $D + \delta$ mit DBSCAN.

Darüber hinaus existiert eine Reihe von weiteren Verfahren, die sich mit dem Problem der Aktualisierung von Cluster-Beschreibungen in spezifischen Anwendungsdomänen befassen. In [Charikar u. a. 1997] wird z. B. eine Methode beschrieben, die insbesondere für den Bereich des Information Retrieval geeignet ist, während in [Aslam u. a. 1999] Lösungen für die inkrementelle

Cluster-Analyse von Dokumenten vorgeschlagen werden.

Alle in diesem Abschnitt vorgestellten Verfahren zur inkrementellen Datenanalyse sind für die effiziente Aktualisierung der gefundenen Muster gut geeignet. Darüber hinaus sind im Abschnitt 3.4 Studien beschrieben, die sich mit dem graduellen Lernen von Klassifikatoren befassen. Problematisch bleibt jedoch, daß das Ziel lediglich in der *Aktualisierung* der Mining-Ergebnisse liegt: Wird festgestellt, daß sich die Muster geändert haben, werden sie zwar aktualisiert, es wird jedoch keine weitere Untersuchung der beobachteten Änderungen vorgenommen. Für den Anwendungsexperten ist es jedoch wichtig zu wissen, *wie* sich die Ergebnisse im Zeitverlauf ändern. Da die Ergebnisse aber über sämtliche Perioden aggregiert werden, kann er die Änderungen nicht nachvollziehen.

## 3.2 Erkennung von Musteränderungen

Während der Fokus inkrementeller Techniken darauf liegt, die Aktualisierung effizient durchzuführen, konzentrieren sich die in diesem Abschnitt vorgestellten Arbeiten auf die *Erkennung* von Änderungen. Dabei ist es in erster Linie nicht immer bezweckt, die Aktualisierung tatsächlich durchzuführen, sondern oftmals geht es nur darum festzustellen, wann dieses erforderlich wäre. Zusätzlich wird auf die Frage eingegangen, von welchen Änderungen die Regeln betroffen sein können.

In [Lee und Cheung 1997] bzw. [Lee u. a. 1998] wird die stichprobenbasierte Methode DELI (Difference Estimation for Large Itemsets) zur Bestimmung des Zeitpunktes, zu dem eine Aktualisierung der gefundenen Muster notwendig ist, vorgeschlagen. Dem Argument folgend, daß bei nur geringen Abweichungen in $\delta$ die Muster in $P$ eine ausreichend gute Approximation darstellen, wird eine Aktualisierung nur dann durchgeführt, wenn der geschätzte Unterschied zwischen den Mustern in $D$ und $\delta$, der aufgrund der Analyse einer Stichprobe aus $D + \delta$ bestimmt wird, dies notwendig erscheinen läßt. Die Entscheidung wird dabei aufgrund einer Maßzahl getroffen, die sich aus dem Quotienten der symmetrischen Differenz der häufigen Itemsets $L^D$ und $L^{D+\delta}$ und der Summe der Anzahl der häufigen Itemsets $|L^D|$ und $|L^{D+\delta}|$ ergibt.[2] Während dieses Maß in [Lee und Cheung 1997] aufgrund der auf der Stichprobe basierenden Analyse-Ergebnisse berechnet wird, werden in [Lee u. a. 1998] Methoden vorgestellt, um untere und obere Grenzen für diesen Wert

[2] Die symmetrische Differenz ergibt sich aus $(L^D - L^{D+\delta}) \cup (L^{D+\delta} - L^D)$.

zu schätzen. Auf diese Weise muß die Stichprobe nicht mehr regelmäßig analysiert werden, sondern nur *bei Bedarf*. Änderungen der Datenbank werden dann so lange gesammelt, bis die Abweichung zwischen $L^D$ und $L^{D+\delta}$ größer als die vom Nutzer vorgegebene Schwelle ist. DELI selbst ist kein Miner – falls eine Aktualisierung von $P$ notwendig wird, wird dafür der Algorithmus $\mathsf{FUP}_2$ verwendet.

In [Pěchouček u. a. 1999] wird das Problem der schrittweisen Evolution von Konfigurationswissen in einer Wissensbasis betrachtet. Es werden zwei Methoden des Maschinellen Lernens, *Inductive Logic Programming* (ILP) und *Explanation Based Generalization* (EBG), im Kontext der Entscheidungsplanung auf ihre Fähigkeit hin verglichen, die Probleme, die sich durch evolvierende Wissensbasen ergeben, zu handhaben. Zum einen wird untersucht, wie die beiden Methoden, die sich durch Repräsentation und Akquisition des Wissens erheblich unterscheiden, mit der Komplexität umgehen, die sich durch den Revisionsprozeß ergibt. Zum anderen wird die Qualität der getroffenen Entscheidungen im Hinblick auf die revidierte Wissensbasis überprüft. Dabei wird unterschieden zwischen *schwachen* Aktualisierungen, die nur die relevanten Teile der Wissensbasis erneut berechnen, und *starken* Aktualisierungen, welche eine Neuberechnung der gesamten Inferenzdatenbank erfordern. Es wird festgestellt, daß die schwache Aktualisierung der Wissensbasis im Falle von ILP aufgrund der verwendeten Wissensrepräsentation sehr gute Ergebnisse liefert: Das Hinzufügen neuer positiver Beispiele ist unkompliziert und führt im Hinblick auf die Anzahl der durchgeführten Aktualisierungen zu einem akzeptablen linearen Effizienzverlust. Darüber hinaus konnte ein Mechanismus gefunden werden, um die Frequenz starker Aktualisierungen in Abhängigkeit von der gewünschten Effizienz des Systems zu schätzen. Im Gegensatz dazu wurde im Fall von EBG festgestellt, daß die für eine schwache Aktualisierung erforderliche Zeit deutlich höher lag als im Fall von ILP. Darüber hinaus ist die Anwendung von EGB auf solche Bereiche begrenzt, in denen menschliche Expertise vorliegt, da die Methode strategisches Inferenzwissen benötigt, um einen anfänglichen Entscheidungsgraphen abzuleiten.

In [Ganti u. a. 1999] wird das Framework `FOCUS` vorgestellt, mit dessen Hilfe der *Unterschied* zwischen zwei Datensätzen bestimmt werden kann. Dazu wird ein Abweichungsmaß berechnet, das den Unterschied in bezug auf das jeweils abgeleitete Modell angibt und deswegen für den Nutzer leicht zu interpretieren ist. Die grundlegende Idee von `FOCUS` ist, das in den Daten gefundene Modell in eine strukturelle Komponente und eine Maßkom-

ponente zu zerlegen. Mit der strukturellen Komponente lassen sich dann die interessanten Regionen des Modells bestimmen, während mit der Maßkomponente jene Teile der Daten zusammengefaßt werden können, die durch die jeweils interessanten Regionen abgebildet werden. Mit Hilfe dieser Zerlegung gelingt es, bestimmte Teile des abgeleiteten Modells gezielt miteinander zu vergleichen. Darüber hinaus definiert `FOCUS` eine Menge von Operatoren, mit denen die Bestandteile der Daten identifiziert werden, die zu interessanten Änderungen im entdeckten Modell geführt haben. Diese Vorgehensweise kann bei einer großen Klasse von Modellen angewandt werden, erwähnt sind z. B. häufige Itemsets, Entscheidungsbaum-Klassifikatoren und Cluster-Beschreibungen. Um den Einfluß von $\delta$ auf $P$ zu ermitteln, muß somit nur noch $\delta$ analysiert und müssen die Ergebnisse mit $P$ verglichen werden. Da $\delta$ nicht notwendigerweise eine Aktualisierung von $D$ darstellen muß, ist der Ansatz von `FOCUS` generischer Natur – prinzipiell ließe sich der Unterschied zwischen zwei beliebigen Datensätzen quantifizieren. Änderungen des Inhalts der entdeckten Muster können grundsätzlich nur im Rahmen einer Mining-Session erkannt werden. Während `FOCUS` jedoch keine temporalen Aspekte beobachteter Änderungen betrachtet, lassen sich die Erwartungen des Anwendungsexperten in Form der früheren Ergebnisse in die Analyse einbeziehen. Für die festgestellten Abweichungen können dann die verursachenden Änderungen in den Daten identifiziert werden.

In [Ganti u. a. 2000] schlägt die gleiche Forschergruppe das Framework DEMON (Data Evolution and MONitoring) vor, das sich mit der Problematik systematischer und nichtsystematischer Aktualisierungen des Datensatzes beschäftigt.[3] Während `FOCUS` im wesentlichen den Inhalt der Muster betrachtet, konzentriert sich DEMON auf die zu untersuchenden Daten. Die Autoren schlagen eine zusätzliche Dimension für die Musterentdeckung vor, die sogenannte *Data Span Dimension*, die es dem Nutzer erlaubt, die Analyse auf eine nach zeitlichen Kriterien begrenzte Untermenge der Datenbank zu beschränken. In ihrer Arbeit plädieren sie für einen generischen Algorithmus, der diesen zusätzlichen Parameter für die Analyse auswertet. Dieser Algorithmus kann für jeden inkrementellen Algorithmus instanziiert werden, um ihn in die Lage zu versetzen, die zusätzliche zeitliche Beschränkung bei der Analyse zu berücksichtigen. Das wird am Beispiel von inkrementellen

[3] Unter systematischen Änderungen wird dabei das reguläre Hinzufügen von Daten im Kontext von Data Warehouses verstanden, während mit nichtsystematischen Änderungen Aktualisierungen im Bereich operationaler Datenbanken gemeint sind.

Erweiterungen für spezifische Algorithmen zur Entdeckung häufiger Itemsets und zur Cluster-Analyse demonstriert. Die ausgewählten Datenblöcke müssen dabei nicht von einheitlicher Größe sein – der Anwendungsexperte erhält die Möglichkeit, lediglich die relevanten Teile der Datenbasis zu analysieren. Die gefundenen Muster erfüllen die statistischen Kriterien der Analyse dann für den jeweils ausgewählten Zeitraum. Die vorgeschlagene Vorgehensweise kann zwar nicht unmittelbar zur Erkennung von Musteränderungen verwendet werden, ihre Anwendung beantwortet aber in jedem Fall die Frage, ob es seit der letzten Mining-Session zu Änderungen der Muster gekommen ist.

In der Studie von Chen und Petrounias [1999] wird dagegen versucht, *Gültigkeitsintervalle* für Assoziationsregeln zu finden. Ausgehend von den gefundenen Regeln, soll bestimmt werden, in welchen Zeitintervallen eine bestimmte Regel gilt und welche Periodizitäten sie aufweist. Zu diesem Zweck wird die Transaktionsdatenbank auf Basis der gespeicherten Zeitstempel der Transaktionen in künstliche Intervalle eingeteilt, die atomar und fixer Länge sind. Für eine gegebene Regel wird anschließend bestimmt, in welchen Intervallen sie zu finden ist. Für die Analyse der gefundenen Intervalle schlagen die Autoren zwei Algorithmen, LISeeker (Longest Interval Seeker) und PIDeriver (Periodicity Interval Deriver), vor. LISeeker wird verwendet, um die längsten, zusammenhängenden Zeitintervalle zu bestimmen, in denen die Regel den Grenzen $\tau_s$ und $\tau_c$ genügt, und PIDeriver wird angewandt, um Zyklen regulärer Intervalle zu ermitteln, in welchen die jeweilige Regel gilt. Im Prinzip ist der Anwender mit der vorgeschlagenen Technik in der Lage festzustellen, zu welchen Zeitpunkten Änderungen in den Daten dazu führen, daß existierende Regeln ungültig werden. Problematisch ist jedoch, daß die Auswertung auf den gefundenen Regeln beruht. So ist es nicht unwahrscheinlich, daß eine periodisch auftretende Regel die Anforderungen bezüglich $\tau_s$ und $\tau_c$ bei Betrachtung des gesamten Datensatzes nicht erfüllt.

Der Vorschlag von Chang u. a. [2002] betrachtet ebenfalls die Frage, in welchem Zeitraum ein Muster gültig ist. Anders als Chen und Petrounias [1999] gehen die Verfasser aber nicht von den entdeckten Regeln aus, sondern bestimmen diese gemeinsam mit ihren Gültigkeitsintervallen. Zu diesem Zweck betrachten sie die Zeiträume, in denen ein Item jeweils in der Transaktionsdatenbank erscheint, und partitionieren die Daten derart, daß für alle in einer Partition enthaltenen Items entweder die Startpunkte oder die Endpunkte ihrer Anwesenheit übereinstimmen. Für die Muster-Entdeckung führen sie das Konzept der *Maximal Common Exhibition Period* (MCP) ein,

das Zeitintervall, in dem alle in einem Itemset enthaltenen Items gemeinsam auftreten. Der vorgeschlagene Algorithmus leitet dann aus jeder Partition Regeln der Form $(X \Rightarrow X)^{[p,q]}$ ab, wobei $[p, q]$ das Zeitintervall bezeichnet, in dem die Regel gilt, d. h. $MCP(X \cup Y) = [p, q]$. Obwohl mit dieser Methode die wichtige Frage nach der Gültigkeit eines Musters beantwortet werden kann, weist sie einen entscheidenden Nachteil auf: Alle statistischen Eigenschaften wie Support oder Konfidenz werden nur noch in bezug auf die MCP bestimmt. Daraus folgt jedoch, daß Muster nur noch dann miteinander verglichen werden können, wenn sie die gleiche MCP aufweisen. Abhängigkeiten zwischen den Mustern wären damit ebenfalls nur noch bei gleicher MCP zu bestimmen, wodurch die Aussagekraft einer derartigen Analyse stark eingeschränkt wird.

## 3.3 Wichtigkeit von Mustern

Ein zentrales Problem, das sich aufgrund der häufig sehr großen Anzahl gefundener Regeln ergibt, ist die Auswahl der Regeln, deren Änderungen verfolgt werden sollen – eine Entscheidung, die in verschärfter Form auch für die beobachteten Regeländerungen getroffen werden muß: Der Nutzer wird im allgemeinen nur an den für die jeweilige Problemstellung relevanten Mustern interessiert sein, eine Aussage, die sicher auch für die Änderungen zutreffen wird.

### 3.3.1 Bewertung von Mustern

Wie bereits im Abschnitt 2.4 angedeutet wurde, gibt es eine große Vielfalt an verschiedenen Maßen, die eine Einschätzung der Qualität der Mining-Ergebnisse ermöglichen [Gago und Bento 1998; Tan und Kumar 2000; Tan u. a. 2002]. Einer weitverbreiteten Meinung zufolge ist die Frage, ob eine gegebene Regel für den betrachteten Anwendungsfall *tatsächlich* von Interesse ist, ausschließlich subjektiv zu beantworten. In [Liu u. a. 1997] wird eine Methode entwickelt, um die subjektive Bewertung der gefundenen Regeln zu formalisieren: Der Anwendungsexperte kodifiziert zunächst sein Hintergrundwissen mit Hilfe einer eigens entworfenen Repräsentationssprache. Anschließend werden die vorgeschlagenen Algorithmen verwendet, um einen Abgleich dieses Wissens mit den gefundenen Regeln durchzuführen. Dabei werden insbesondere jene Regeln identifiziert, die unerwartet und folglich interessant für den

Anwender sind.

In [Freitas 1998] wird dem Primat subjektiver Maße widersprochen: Neben den subjektiven Maßen, die im wesentlichen durch den Nutzer bestimmt sind, gibt es auch eine Notwendigkeit für objektive Maße. Der Vorzug dieser durch die Daten bestimmten Maße besteht darin, daß sie generischer sind und damit nicht so stark vom Anwendungskontext abhängen. Im Kontext von Klassifikationsregeln schlägt der Autor unter anderem ein Bewertungsmaß vor, das auf der *Umkehrung* konventioneller Maßstäbe beruht. Er argumentiert, daß Regeln, deren linke Seite nur eine geringe Zahl von Beispielen abdecken, für den Nutzer besonders interessant sind, da sie häufig unerwartete Zusammenhänge in den Daten beschreiben. Solche Regeln seien zwar fehleranfällig und weniger generell als häufige Regeln, werden sie aber unbegründet verworfen, kann die Klassifikationsgenauigkeit in erheblichem Maße zurückgehen.

Ebenfalls im Bereich der Klassifikation ist die in [Freitas 1999] vorgestellte Studie angesiedelt, in der verschiedene Faktoren diskutiert werden, die beim Entwurf von Bewertungsmaßen berücksichtigt werden müssen. So wird unter anderem das Problem der ungleichmäßigen Verteilung der Klassen beschrieben: Ist die Anzahl der Beispiele für eine der Klassen wesentlich größer, so fällt es leicht, Regeln zu finden, die diese Klasse beschreiben. Andererseits gelingt es ungleich schwerer, Regeln zu finden, durch die Klassen mit nur wenigen Beispielen beschrieben werden – ein Problem, das mit den in [Freitas 1998] besprochenen Fragen verwandt ist. Vor dem Hintergrund dieser Diskussion wird anschließend ein Vorschlag gemacht, wie ein Qualitätsmaß unter Einbeziehung der angesprochenen Faktoren entworfen werden könnte.

Alle bisherigen Arbeiten betrachten ausschließlich statische Aspekte des entdeckten Wissens. Entweder auf subjektiven oder objektiven Kriterien basierend, werden die Eigenschaften der Regeln zeitpunktbezogen ausgewertet, um eine Entscheidung bezüglich der Relevanz der Regel für eine bestimmte Anwendungsdomäne zu treffen. Diesem Muster folgend, ergäbe sich für die Beobachtung der Regeln folgendes Schema: Nachdem mit einem oder mehreren der hier oder im Abschnitt 2.4 erwähnten Verfahren eine Menge von interessanten bzw. relevanten Mustern ausgewählt wurde, können die Änderungen dieser Muster in den folgenden Mining-Sessions aufgezeichnet und überwacht werden. Natürlich kann eine solche Vorgehensweise begründet sein, es könnte dabei jedoch auch festgestellt werden, daß die ausgewählten Muster keine oder nur sehr geringe Änderungen im Zeitverlauf zeigen. Andererseits könnten sich Muster, die den verwendeten Kriterien nicht genügen,

in einer Weise ändern, die für die betrachtete Anwendung von großem Interesse wäre. In [Liu u. a. 2001b] wird ein erster Vorschlag gemacht, wie sich die Änderungen, denen eine Regel im Zeitverlauf ausgesetzt sein kann, in die Bewertung der Wichtigkeit eines Musters einbeziehen lassen. Es wird die Frage untersucht, ob der Nutzer den gefundenen Regeln vertrauen kann. Dazu wird zwischen stabilen Regeln, die keine deutlichen Schwankungen zeigen, Regeln, die einen Trend zeigen, und semi-stabilen Regeln unterschieden. Der Datenbestand wird in Partitionen geteilt, welche separat analysiert werden. Mit Hilfe verschiedener statistischer Testverfahren werden dann die Statistiken der gefundenen Regeln untersucht, um sie den unterschiedlichen Gruppen zuzuordnen.

Obwohl nur eine sehr begrenzte Menge verschiedener Änderungen betrachtet wird und die gefundenen Änderungen selbst nicht weiter untersucht werden, sind in dieser Arbeit erstmals temporale Aspekte in die Bestimmung der Wichtigkeit der Muster eingeflossen. Bei den verwendeten Testverfahren ist es jedoch notwendig, für alle Regeln, die in mindestens einer Periode gefunden werden konnten, die Statistiken auch für all jene Perioden zu berechnen, in denen sie *nicht* gefunden wurden. Ein grundsätzliches Problem betrifft jedoch die Auswahl der *interessanten* Regeln: Zum einen können – das Argument in [Freitas 1998] auf die temporale Dimension übertragend – für den Nutzer auch jene Muster interessant sein, die sich nur selten in den Daten manifestieren, da sie lediglich Ausnahmen signalisieren. Zum anderen könnte ein Muster, das in allen Partitionen stabile Statistiken *unterhalb* der Schwellen $\tau_s$ oder $\tau_c$ zeigt und diese nur in einer einzelnen Partition überschreitet, in die Menge der stabilen Muster aufgenommen werden, obwohl es eigentlich weder aus statischer noch aus dynamischer Sicht wirklich interessant ist.

### 3.3.2 Bewertung von Musteränderungen

Die in [Liu u. a. 2001b] beschriebene Arbeit betrachtet zwar die Dynamik von Regeln und prüft für die Zuordnung der entdeckten Regeln zu den verschiedenen Gruppen, ob sie im Zeitverlauf Änderungen zeigen, es wird jedoch keine Analyse der beobachteten Änderungen vorgenommen: Der Anwender erhält lediglich die Information, *daß* sich die Regel geändert hat. Eine naheliegende Vorgehensweise bestünde darin, die in den verschiedenen Perioden ermittelten Werte der statistischen Eigenschaften miteinander zu vergleichen. Übersteigt die relative oder absolute Abweichung zwischen zwei Perioden ei-

ne vorgegebene Grenze, wird sie als interessant gekennzeichnet. Wenn auch mit der anwendungsspezifischen Kombination von relativen und absoluten Grenzen im Hinblick auf die jeweilige Anwendung relativ gute Resultate zu erzielen sind, ist die Anzahl der signalisierten interessanten Änderungen mitunter noch sehr hoch. In [Agrawal u. a. 1995] wird mit einer SQL-ähnlichen Anfragesprache versucht, Historien bestimmter Gestalt zu finden. Dazu gibt der Nutzer eine Schablone für die gesuchte Historie vor, für die dann in den Zeitreihen der Musterstatistiken Instanzen gesucht werden. Dieser Ansatz hat den Nachteil, daß der Nutzer bereits über Vorstellungen bezüglich interessanter Historien verfügen muß, um die Schablone definieren zu können. In [Ganti u. a. 1999] wurde versucht, aufgrund eines Vergleichs der interessanten Regionen des abgeleiteten Modells auf relevante Änderungen zu schließen. Dieser Ansatz konzentriert sich jedoch darauf, die Daten zu identifizieren, die zu den interessanten Änderungen im Modell geführt haben (vgl. Abschnitt 3.2).

Die in [Dong und Li 1999] beschriebene Studie schlägt einen Algorithmus zur Entdeckung von Mustern vor, deren Support zwischen zwei Mining-Sessions stark ansteigt, sogenannte *Emerging Patterns*. Prinzipiell werden hier ausschließlich Muster extrahiert, die interessante Änderungen zeigen, wobei der Begriff des *Interesses* ausschließlich aufgrund von Änderungen des Supports definiert wird. Besonderes Augenmerk wird in dieser Arbeit auf die effiziente Gestaltung des Algorithmus gelegt, da das Prinzip des *Apriori*-Algorithmus in diesem Fall nicht anwendbar ist und die Anzahl und Länge von Itemsets mit niedrigem Support sehr hoch sein kann.

In [Liu u. a. 2001a] wird die Frage betrachtet, welche Regeländerungen *fundamentaler* Natur sind und dem Nutzer gemeldet werden sollten. Der Grundgedanke ist dabei, daß nur solche Regeländerungen interessant sind, die nicht auf Änderungen anderer Regeln zurückgeführt werden können. Um die fundamentalen Änderungen zu bestimmen, betrachten die Autoren den Inhalt von Regeln der Form $r_1, \ldots, r_{m-1} \rightarrow r_m$ und überprüfen Änderungen des Supports und der Konfidenz der Regeln $r_i \rightarrow r_m$, $i \leq m$ zwischen zwei Zeitpunkten mit Hilfe des $\chi^2$-Tests. Diese Vorgehensweise ist durchaus gut geeignet, eine signifikante Änderung der jeweiligen statistischen Maßzahl zwischen den beiden betrachteten Perioden zu ermitteln. Sollen jedoch Änderungen langfristiger Natur ermittelt werden, ist dieses Mittel inadäquat, da immer nur zwei Perioden einbezogen werden.

In [Au und Chan 2002] wird ein unscharfer Ansatz verwendet, um interessante Änderungen in Assoziationsregeln zu entdecken. Auch bei dieser Methode ist es zunächst notwendig, die Statistiken für die Muster in allen Pe-

rioden zu ermitteln. Aus den Zeitreihen der Muster werden anschließend mit Hilfe des vorgestellten Algorithmus unscharfe Regeln extrahiert, die in den Folgeperioden zur Vorhersage der Entwicklung der statistischen Eigenschaften der Muster benutzt werden können. Dabei repräsentieren die abgeleiteten unscharfen Regeln die Regelmäßigkeiten in der Entwicklung der Musterstatistiken. Ob eine gegebene Änderung tatsächlich interessant ist, bestimmen die Autoren mit Hilfe eines objektiven Maßes, welches auf der Grundlage von Signifikanztests ermittelt wird.

Eine völlig andere Herangehensweise schlagen Chakrabarti u. a. [1998] vor. Ihr Ansatz beruht nicht darauf, *erst* eine Menge von Mustern zu entdecken, um interessante Änderungen für sie zu ermitteln. Statt dessen sucht der vorgeschlagene Algorithmus ausschließlich nach jenen Mustern, die interessante Änderungen im Zeitverlauf zeigen. Das Interesse wird in diesem Fall mit Hilfe einer speziellen Kodierung der Itemsets bestimmt, die auf der Korrelation zwischen den Items beruht. Dem Argument folgend, daß eine gleichbleibende Korrelation zwischen den Items bekannt und somit wenig interessant sein kann, ist das Kodierungsschema so entworfen, daß eine vergleichsweise kleine Anzahl von Bits benötigt wird, um Itemsets zu kodieren, deren Items stationäre Korrelationen aufweisen, und eine große Anzahl von Bits, um Itemsets zu kodieren, deren Items stark schwankende Abhängigkeiten aufweisen. Welche Itemsets aufgrund ihrer Änderungen in der Zeit interessant sind, kann dann durch einen einfachen Vergleich der Kodierungslänge ermittelt werden. Hervorzuheben ist, daß dieser Ansatz völlig ohne die sonst im Bereich der Warenkorbanalyse üblichen Beschränkungen in der Mining-Anfrage $\tau_s$ und $\tau_c$ auskommt. Zusätzlich wird eine Partitionierung der Zeitachse vorgeschlagen: Basierend auf der Kodierungslänge, werden die Partitionen jeweils so gewählt, daß die Unterschiede zwischen aufeinanderfolgenden Perioden maximal sind. Der Nachteil besteht allerdings darin, daß die Partitionierung der Zeitachse von der jeweiligen Regel abhängt und ein direkter Vergleich der Entwicklung von Regeln nicht mehr möglich ist.

## 3.4 Konzeptdrift

Musteränderungen, ob in bezug auf den Inhalt der Muster oder ihre statistischen Eigenschaften, können temporärer Natur sein oder durch grundlegende Änderungen in der betrachteten Population verursacht sein. Änderungen grundsätzlicher Natur, die sich dauerhaft oder zumindest für einen gewissen

Zeitraum in den abgeleiteten Modellen widerspiegeln, werden im allgemeinen unter dem Begriff der Konzeptdrift zusammengefaßt. Insbesondere im Hinblick auf die Bewertung von Änderungen ist dieser Begriff natürlich auch im Rahmen der vorliegenden Arbeit von Interesse. Probleme, die sich durch Konzeptdrift ergeben, sind vor allem im Maschinellen Lernen Gegenstand intensiver Forschung, und insbesondere im Bereich der Klassifikation gibt es eine Vielzahl von Studien aus unterschiedlichen Anwendungsbereichen, von denen im folgenden einige beschrieben werden sollen.

In [Morik und Rüping 2002] wird Konzeptdrift im Kontext von ILP untersucht, die zur Bestimmung der Phasen des Konjunkturzyklus eingesetzt wird. Die Studie in [Klinkenberg und Joachims 2000] beschreibt eine Methode zur adaptiven Text-Klassifikation mit Hilfe von *Support Vector Machines* (SVMs). Grundgedanke ist auch hier die verbreitete Idee, nur die Daten eines bestimmten Zeitfensters zum Lernen des Klassifikators zu verwenden [Widmer und Kubat 1996]. In diesem Fall wird die Fenstergröße jeweils so gewählt, daß der geschätzte Generalisierungsfehler minimiert wird. Auch in [Syed u. a. 1999] werden SVMs zum inkrementellen Lernen von Klassifikatoren verwendet. Zusätzlich schlagen die Autoren Kriterien zur Einschätzung von Robustheit und Zuverlässigkeit des gelernten Klassifikators vor.

In [Grieser 2000] wird eine Methode zum inkrementellen Lernen von Entscheidungsbaum-Klassifikatoren vorgestellt, die einen limitierten Speicher für die Instanzen verwendet. Durch diese Beschränkung wird immer nur eine begrenzte Anzahl von Beispielen in die Hypothesenbildung einbezogen. Im Unterschied zu anderen Methoden, bei denen die Bildung von Hypothesen nur auf Basis der neuen Daten vorgenommen wird, werden in diesem Fall zusätzlich die zwischengespeicherten Instanzen und die alten Hypothesen genutzt. Im Kontext der Verarbeitung permanenter Datenströme stellen Hulten u. a. [2001] ebenfalls eine Methode zum Lernen von Entscheidungsbäumen vor, die in der Lage ist, sich fortlaufend ändernden Konzepten anzupassen. Die Besonderheit des vorgeschlagenen Algorithmus CVFDT (*Concept-adapting Very Fast Decision Tree Learner*) liegt darin, daß er einen zusätzlichen Unterbaum anlegt, wenn die Präzision des aktuellen Baumes bei der Klassifikation neuer Instanzen zurückgeht. Weist der neue Baum eine höhere Klassifikationsgüte auf, wird der alte Baum durch ihn ersetzt.

Ein völlig anderer Weg wird in [Street und Kim 2001] beschritten. Hier werden mehrere separate Klassifikatoren auf sequenziellen Datenblöcken trainiert, die dann gemeinsam in einem *Ensemble* fixer Größe benutzt werden. Welche Kombination von Klassifikatoren jeweils Verwendung findet, wird mit

Hilfe einer Heuristik ermittelt, die solche Entscheidungsbäume bevorzugt, die in Fällen, in denen das Ensemble keine eindeutige Eintscheidung treffen konnte, die richtige Klasse bestimmt haben.

Der Frage, inwieweit akkurates Lernen zum Zwecke der Vorhersage überhaupt möglich ist, wenn in der betrachteten Datensequenz Konzeptdrift auftritt, wird in [Case u. a. 2001] nachgegangen. Hierbei werden verschiedene Maße, die für die Bewertung der Vorhersagequalität geeignet sind, vorgestellt und verglichen.

Oft werden im Rahmen des Anwendungskontextes Annahmen gemacht oder gegebene externe Faktoren einbezogen, die zwar in den Lernprozeß einfließen, für den Nutzer aber häufig nicht offensichtlich sind – ein Problem, das auch in [Widmer und Kubat 1996] diskutiert wird. Der sogenannte *verborgene Kontext* kann die Qualität des Klassifikators deutlich beeinträchtigen, insbesondere wenn im Zeitverlauf unbemerkt Kontextänderungen auftreten. Der in dieser Arbeit vorgeschlagene Algorithmus begegnet dem Problem der Konzeptdrift mit der bereits beschriebenen Methode, ein gleitendes Zeitfenster zu verwenden, aus dem die Instanzen für den Lernprozeß gewonnen werden. Um Kontextänderungen in der Anwendung zu erkennen, verfolgen die Autoren folgenden Ansatz: Wird vermutet, daß ein Kontextwechsel stattfindet, werden die gespeicherten Konzeptbeschreibungen konsultiert, um festzustellen, ob sie die Beispiele, die sich im aktuellen Zeitfenster befinden, besser erklären. Umgekehrt werden *stabile* Hypothesen in den Zwischenspeicher übertragen, um sie für den Fall, daß der gegenwärtige Kontext erneut auftritt, verfügbar zu machen. Dazu überprüft die Heuristik, die verwendet wird, um die Fenstergröße anzupassen, bei der Verarbeitung eines neuen Beispiels, ob die aktuelle Hypothese stabil ist. In diesem Fall wird die aktuelle Hypothese gespeichert, es sei denn, es existiert bereits eine Hypothese, die die gleiche Menge positiver Beispiele betrifft. Ist sie dagegen nicht stabil, was sich durch eine Verkleinerung des Zeitfensters bemerkbar macht, wird die aktuelle Hypothese mit den gespeicherten Hypothesen verglichen und die beste ausgewählt.

Ein weiteres Problem, das häufig in diesem Zusammenhang erwähnt und teilweise als Ursache für Konzeptdrift anzusehen ist, wird mit Populationsdrift bezeichnet [Kelly u. a. 1999]. Hierbei geht es nicht um Änderungen in den abgeleiteten Modellen, sondern um Änderungen, die die jeweils untersuchte Population betreffen. In ihrer Studie untersuchen die Autoren die Auswirkungen der möglichen Änderungen, mit denen im Rahmen von Populationsdrift zu rechnen ist. Im einzelnen diskutieren sie die Folgen von Änderungen der A-priori-Wahrscheinlichkeiten der Klassen $P(K_i)$, der Ver-

teilung in den Klassen $P(x|K_i)$ und der A-posteriori-Wahrscheinlichkeiten für die Klassenzugehörigkeit $P(K_i|x)$. Zusätzlich wird eine Methode beschrieben, die Klassifikationsregel in ein größeres, allgemeineres Modell zu integrieren, das eine evolvierende Population toleriert.

Adaptive Klassifikationsalgorithmen bestimmen den Einfluß jedes neuen Beispiels auf den Klassifikator, um ihn bei Bedarf entsprechend anzupassen. Damit sind adaptive Methoden im Prinzip in der Lage, mit Problemen wie Konzept- oder Populationsdrift umzugehen. Ist Adaptivität jedoch in jedem Fall das geeignete Mittel? Wenn z. B. die Frequenz neuer Einträge im Transaktionsprotokoll sehr hoch ist oder einem einzelnen Eintrag ein nur sehr geringes Gewicht zukommt, wäre die Berücksichtigung jedes einzelnen Beispiels ineffizient. Aber auch die Frage, ob Adaptivität in diesem Maße wirklich *gewollt* ist, muß bedacht werden: Oft wird Konzeptdrift durch Änderungen externer Faktoren hervorgerufen, und in einem solchen Fall ist es von Interesse, die Ursache der Änderung zu identifizieren oder zumindest den Zeitpunkt bzw. Zeitraum ihres Einflusses zu ermitteln. Zu diesem Zweck wäre es jedoch notwendig, die Änderungen als diskrete Ereignisse wahrzunehmen, die eindeutigen Zeitpunkten oder -intervallen zugeordnet sind. Da Konzeptdrift ein Phänomen ist, das sich oft langsam und über einen längeren Zeitraum manifestiert, könnte ein adaptiver Ansatz dies nicht leisten. Diese Fragen sind auch bei der Anwendung inkrementeller Verfahren zur Cluster-Analyse relevant.

## 3.5 Trendanalyse

Auch auf dem Gebiet der Trendanalyse, die sich mit der Beobachtung der Entwicklung von Variablen im Zeitverlauf befaßt, gibt es eine Vielzahl relevanter Studien. Dabei verdienen insbesondere solche Arbeiten Beachtung, die sich mit der Erkennung bzw. Beobachtung lokaler Extrema und Trends beschäftigen und Gemeinsamkeiten zwischen der zeitlichen Entwicklung unterschiedlicher Variablen finden können. Im allgemeinen arbeiten Verfahren der Trend- oder Zeitreihenanalyse dabei auf *Primärdaten*, d. h. auf direkten Beobachtungen der betrachteten Population. Im Rahmen der vorliegenden Arbeit sind dagegen eher Methoden relevant, die sich auf die Analyse von Sekundärdaten konzentrieren, da sich die Beobachtungen auf die aus den Daten extrahierten Muster und Modelle beziehen. Natürlich bestehen viele Gemeinsamkeiten zwischen der Analyse von Primär- und Sekundärdaten,

einige Konzepte lassen sich jedoch nicht ohne weiteres übertragen – zum einen, weil zugrunde liegende Annahmen verletzt würden, zum anderen aber auch, weil die Effekte konventioneller Zeitreihenanalyse nicht gewollt wären. Eine übliche Annahme bei der Trendanalyse ist z. B., daß sich die beobachtete Population nicht oder nur sehr wenig ändert – eine Voraussetzung, von der bei der Analyse von Sekundärdaten nicht immer ausgegangen werden kann. Ein anderes Beispiel betrifft Verfahren, die zur Ermittlung von Trends eine Glättung der beobachteten Zeitreihen vornehmen: Handelt es sich bei einem Ausreißer tatsächlich nur um Rauschen in den Daten, oder stellt er ein möglicherweise interessantes Phänomen dar?

Im Bereich der Analyse von Zeitreihen gibt es in dem hier betrachteten Kontext zwei grundsätzlich unterschiedliche Methoden. Zum einen wird angestrebt, im voraus bekannte Muster in den Zeitreihen zu finden – ein Ansatz, der unter anderem in [Agrawal u. a. 1995] beschrieben ist. Wie bereits erwähnt, wird in dieser Arbeit mit einer SQL-ähnlichen Anfragesprache versucht, Zeitreihen bestimmter *Form* zu finden (vgl. Abschnitt 3.3.2). Das Problem bei dieser Vorgehensweise ist jedoch, daß der Anwendungsexperte die Gestalt der Historie bereits kennen muß.[4] Der andere Ansatz, häufig als *Time Series Mining* bezeichnet und eher dem Bereich des Unsupervised Learning zuzuordnen, macht keinerlei Annahmen bezüglich der zu findenden Muster.

In [Das u. a. 1998] ist ein Verfahren beschrieben, um Muster zu finden, die verschiedene Abschnitte der betrachteten Zeitreihen zueinander in Beziehung setzten. Dabei werden zwei Fälle unterschieden: Im ersten geht es um Beziehungen zwischen verschiedenen Abschnitten der gleichen Zeitreihe, im zweiten Fall um Abschnitte unterschiedlicher Zeitreihen. In beiden Fällen sind die Abschnitte nicht vorgegeben, sondern sollen ebenfalls aufgrund der Daten ermittelt werden. Um die Komplexität zu vermeiden, die sich bei der Analyse von stetigen Variablen ergibt, muß dazu eine Diskretisierung der Zeitreihen vorgenommen werden. Die vorgeschlagene Methode verwendet zunächst ein gleitendes Zeitfenster, um die Zeitreihen zu zerlegen. Anschließend werden die resultierenden Subsequenzen mit einer Cluster-Analyse untersucht und die Subsequenzen der Zeitreihe durch die dazugehörigen Cluster-Kennungen ersetzt. Die resultierende diskrete Zeitreihe kann dann mit konventionellen Mining-Techniken analysiert werden, um Assoziationsregeln oder häufige Se-

[4] Andererseits liefert diese Art der Auswertung ausschließlich *interessante* Muster zurück.

quenzen abzuleiten.

In [Lin u. a. 2002] und [Patel u. a. 2002] wird neben einem effizienten Algorithmus zur Suche häufiger Muster eine weitere Methode zur Diskretisierung von Zeitreihen vorgeschlagen, die auf abschnittsweiser Aggregat-Approximation beruht.[5] Vergleichbar mit einem auf Zeitfenstern basierenden Ansatz, wird die Zeitreihe in Intervalle eingeteilt und für jedes Intervall ein Aggregat berechnet, z. B. der Mittelwert der Zeitreihe, und das gesamte Intervall durch diesen einzelnen Wert approximiert. Anschließend wird der Wertebereich der resultierenden Treppenfunktion in Intervalle eingeteilt, und es werden die Mittelwerte durch das mit dem jeweiligen Intervall assoziierte Symbol ersetzt. Neben der Möglichkeit, sehr große Zeitreihen in eine diskrete Repräsentation zu überführen, hat die aufgezeigte Methode die Eigenschaft, daß trotz der diskreten Repräsentation eine nach unten begrenzte Approximation der euklidischen Distanz angegeben werden kann.

Keogh u. a. [2002] befassen sich mit der Suche nach *überraschenden* Mustern in Zeitreihen, wobei sie jene Muster für überraschend halten, die häufiger auftreten als erwartet. Hierbei vergleichen sie die Frequenz, mit der ein Muster in unterschiedlichen Zeitreihen auftritt. Den Ausgangspunkt bildet eine Zeitreihe, die nach Ansicht des Nutzers den *Normalfall* darstellt. Für eine zweite Zeitreihe wird dann mit dem vorgeschlagenen Algorithmus überprüft, ob es überraschende und somit interessante Unterschiede bei den auftretenden Mustern und der Häufigkeit ihres Auftretens gibt. In gewisser Weise ist dieser Ansatz mit der Arbeit von Chakrabarti u. a. [1998] vergleichbar, auch wenn er hier in einer anderen Domäne angewendet wird. Demnach sind alle Muster, die von den Erwartungen des Anwendungsexperten abweichen, von besonderem Interesse für ihn. Welche Muster das im Einzelfall sind, wird ausschließlich anhand der Unterschiede zwischen den beiden betrachteten Datensätzen bestimmt. Liegt keine Referenz-Zeitreihe vor, können ihre Eigenschaften – im Unterschied zur Arbeit von Chakrabarti u. a. [1998] – auch mit Hilfe eines Markow-Modells geschätzt werden. Das Problem der Diskretisierung der Zeitreihen wird in ähnlicher Weise gelöst wie in [Lin u. a. 2002] bzw. [Patel u. a. 2002], so daß der Algorithmus, der für die Suche der Unterschiede verwendet wird, auf einer Sequenz von Symbolen eines endlichen Alphabets arbeitet. Diese Sequenz wird anschließend in einer Indexstruktur abgelegt, die dem Vorschlag von Wang und Tan [1996] sehr ähnlich ist. Unterschiede zwischen den Zeitreihen können dann durch einen einfachen Vergleich

[5] Eigentlich handelt es sich bei den beiden Artikeln um denselben Beitrag.

ihrer Baumrepräsentation ermittelt werden.

Verfahren wie das von Agrawal u. a. [1995] sind immer dann gut geeignet, wenn der Anwender im voraus weiß, welche Muster für die betrachtete Anwendung von Interesse sind. Liegt solches Hintergrundwissen nicht vor, sind die vorgestellten Verfahren zur Entdeckung von zuvor unbekannten Mustern besser geeignet. Die beschriebenen Verfahren arbeiten jedoch alle nach dem gleichen Prinzip: Die Zeitreihe wird in eine diskrete Repräsentation überführt, welche anschließend nach Mustern durchsucht wird. Bei der Betrachtung von sehr langen Zeitreihen sind diese Schritte notwendig, um den Suchraum für die Musterentdeckung überschaubar zu halten. Problematisch ist jedoch, daß mit der Diskretisierung auch ein Informationsverlust einhergeht. Haben diese Informationen im betrachteten Anwendungskontext jedoch eine wesentliche Bedeutung, sind die Verfahren nur bedingt geeignet.

Während sich die Trendanalyse spezifisch der Suche bzw. Entdeckung von Mustern in Zeitreihen widmet, sind unter dem Begriff *Temporal Data Mining* solche Arbeiten zusammengeführt, in denen es um die Analyse von Daten mit einer expliziten zeitlichen Dimension geht. Wie die in [Roddick und Spiliopoulou 1999] vorgenommene Eingrenzung dieses Begriffs zeigt, zählen insbesondere solche Verfahren dazu, die sich mit der Entdeckung kausaler Beziehungen zwischen Ereignissen mit zeitlicher Ordnung befassen.[6] Ein zweiter Aspekt bezieht darüber hinaus auch Arbeiten ein, die sich der Entdeckung ähnlicher bzw. gleicher Muster in Zeitreihen widmen. Vielleicht mit Ausnahme der Arbeit von Agrawal u. a. [1995] fallen damit alle der in diesem Abschnitt beschriebenen Beiträge unter diese Definition, und insofern stellt Temporal Data Mining eigentlich den Oberbegriff für diese Arbeiten dar. Wie der erste Teil der Einordnung zeigt, fallen aber auch viele der Arbeiten, die in den anderen Abschnitten dieses Kapitels diskutiert wurden, in diesen Bereich.

[6] In diesem Beitrag bzw. in der aktualisierten Fassung in [Roddick u. a. 2001] ist auch eine umfangreiche Bibliographie für den Bereich Temporal Data Mining zu finden.

# Kapitel 4

# Verwaltung des entdeckten Wissens

## 4.1 Einführung

Daten werden heute in den verschiedensten Bereichen und für die unterschiedlichsten Zwecke gesammelt. Dabei ist fast allen Datensätzen gemeinsam, daß sie eine zeitliche Dimension aufweisen, Während dieser Aspekt oft direkt im verwendeten Datenmodell erfaßt ist, wird er manchmal nur implizit deutlich, z. B. durch den Zeitraum, in dem der Datensatz gesammelt wurde. Ein Datensatz, der einen bestimmten Ausschnitt der Realität repräsentiert, unterliegt den gleichen Änderungen wie die Realität selbst, da sich Änderungen der Realität in den gesammelten Daten niederschlagen. Das gleiche gilt für die Muster, die in den Daten entdeckt wurden. Auch sie gelten möglicherweise nur für einen bestimmten Zeitraum. Aus diesem Grund dürfen Daten eines längeren Zeitraums bei der Analyse nicht als statische Einheit betrachtet werden. Will man nicht nur erfahren, welche Muster in den Daten enthalten sind, sondern auch ihre zeitliche Gültigkeit und Änderungen erkennen, darf man folglich die temporale Komponente nicht vernachlässigen.

In dieser Arbeit wird Wissensentdeckung als *kontinuierlicher* Prozeß gesehen, der aus wiederholten *Mining-Sessions* besteht. Zu bestimmten Zeitpunkten $t_i$ findet eine Mining-Session statt, in der ein Datensatz $D_i$ analysiert wird, welcher über den Zeitraum $t_i - t_{i-1}$ gesammelt wurde. In jeder Sitzung werden Muster entdeckt, von denen ein Teil bereits aus früheren Sitzungen bekannt, ein anderer Teil neu ist. Alle Muster werden in einer *Regelbasis*

gespeichert, und während neue Muster in diese aufgenommen werden, sind bereits gespeicherte Muster zu aktualisieren. Dabei sind grundsätzlich zwei Komponenten eines Musters zu betrachten: sein Inhalt, d. h. der Zusammenhang in den Daten, den ein Muster beschreibt, und seine statistischen Eigenschaften, die Aussagen über die Gültigkeit eines Musters erlauben. Nur wenn diese beiden Aspekte gemeinsam betrachtet werden, ist es möglich, die Änderungen eines Musters über die Zeit zu erkennen.

In diesem Kapitel sollen die Grundlagen für die Erkennung von Musteränderungen und ihre Analyse dargelegt werden. Im ersten Teil wird das temporale Regelmodell vorgestellt, das als Basis für die Speicherung von Mustern unter Einbeziehung ihrer zeitlichen Dimension dient. Im zweiten Teil wird unter Verwendung dieses Datenmodells gezeigt, auf welche Weise die Aktualisierung der entdeckten Muster vorgenommen werden kann.

## 4.2 Speicherung entdeckter Muster

Um auch die temporale Dimension der Muster zu repräsentieren, findet ein Datenmodell Verwendung, das aus einer statischen Komponente besteht, die sich im Zeitverlauf nicht ändert, und einer dynamischen Komponente, welche die veränderlichen Bestandteile des Musters repräsentiert. Die statische Komponente enthält dabei den Inhalt der Regel, d. h. den Zusammenhang in den Daten, den das Muster beschreibt, während die dynamische Komponente die Zeitreihen der statistischen Eigenschaften des Musters enthält, die im Zeitverlauf fortgeschrieben werden.

### 4.2.1 Das temporale Regelmodell

Das vorgeschlagene Regelmodell stellt ein Muster $P$ als temporales Objekt mit der folgenden Signatur dar [Baron und Spiliopoulou 2001]:

$$P = ((ID, query, body, head), \{(timestamp, statistics)\})$$

In diesem Modell ist $ID$ eine automatisch generierte Kennung, die sicherstellt, daß für alle Regeln, die den gleichen Inhalt haben, auch die gleiche Kennung benutzt wird. Sie findet Verwendung, um ein Muster über den *gesamten* Analysezeitraum hinweg eindeutig zu identifizieren. In *query* ist die Mining-Anfrage gespeichert, die bei der Musterentdeckung verwendet wurde.

Die Speicherung der Anfrage ist wichtig, da die Vergleichbarkeit der Ergebnisse und damit die Möglichkeit zur Fortschreibung der Werte für die erfaßten statistischen Eigenschaften nur sinnvoll ist, solange sich die Mining-Anfrage nicht ändert. In *body* und in *head* ist der Zusammenhang, den das Muster beschreibt, gespeichert.[1]

Alle bisher beschriebenen Bestandteile des Modells sind invariant und bilden die statische Komponente des Modells. Die statistischen Eigenschaften des Musters und ihre zeitliche Entwicklung sind im zweiten Teil des Modells repräsentiert. Diese werden im Zeitverlauf fortgeschrieben und bestehen aus einem Zeitstempel (*timestamp*), dem eine Menge von statistischen Werten (*statistics*) zugeordnet ist.

**Beispiel 4.1** Über einen Zeitraum von mehreren Monaten werden die Zugriffe auf einen Web-Server in einem Transaktionsprotokoll aufgezeichnet. Dabei wird für jeden Nutzer protokolliert, welche Seiten er innerhalb einer Sitzung aufgerufen hat. Eine auf Grundlage des Protokolls durchgeführte Warenkorbanalyse soll zeigen, welche Seiten typischerweise gemeinsam in einer Sitzung aufgerufen werden. Um die Entwicklung dieser Zugriffsmuster im Zeitverlauf verfolgen zu können, wird das Protokoll auf monatlicher Basis geteilt und die Analyse auf den resultierenden Partitionen durchgeführt. Für die Entdeckung von Assoziationsregeln besteht eine Mining-Anfrage im allgemeinen aus der Angabe von unteren Grenzen für den Support und die Konfidenz. Im betrachteten Fall soll zur Vereinfachung davon ausgegangen werden, daß nur ein minimaler Support von $s = 5\%$ zur Musterentdeckung verwendet wurde.

Ein in der initialen Mining-Session $t_0$ gefundenes Muster $R$ mit dem Inhalt „p32.htm $\Rightarrow$ p43.htm“, das einen Support von 9% aufweist, würde gemäß dem temporalen Regelmodell durch

$R_0 = ((\text{“p32:p43”}, s = 5\%, \text{“p32.htm”}, \text{“p43.htm”}), \{(t_0, s = 9\%)\})$

dargestellt. Im Rahmen der nächsten Mining-Session wird festgestellt, daß der Support des Musters auf 7% gefallen ist. Infolge dieser Änderung wird das Muster in der Regelbasis aktualisiert und nach der Änderung durch

$R_1 = ((\text{“p32:p43”}, s = 5\%, \text{“p32.htm”}, \text{“p43.htm”}), \{(t_0, s = 9\%), (t_1, s = 7\%)\})$

dargestellt. ◇

[1] Ursprünglich wurde das Modell für Assoziationsregeln, Implikationen der Form „*Wenn Voraussetzung, dann Ergebnis*“, entwickelt. Im Englischen wird die Voraussetzung einer solchen Regel häufig als *Body* und das Ergebnis als *Head* bezeichnet (vgl. Abschnitt 2.2).

Bei den statistischen Eigenschaften eines Musters ist jedoch zu beachten, daß sie nicht nur für das Muster selbst, sondern normalerweise auch für die *Elemente* der Regel statistische Eigenschaften erfaßt werden. So wird im eben betrachteten Beispiel auch für die Seiten, die im Zugriffsmuster enthalten sind, der Support erfaßt, d. h. der prozentuale Anteil der Sessions, in denen auf die jeweilige Seite zugegriffen wurde, im Gegensatz zum Muster, dessen Support den Anteil der Sessions angibt, in denen auf *beide* Muster zugegriffen wurde. Dieses Problem wird gelöst, indem die Bestandteile des Modells *body* und *head* ebenfalls als Objekte betrachtet werden. Auf diese Weise können sie gemeinsam mit ihren statistischen Eigenschaften erfaßt werden. Ähnliches gilt auch für die Periode. Hier muß zumindest erfaßt werden, wie groß die Population in der betrachteten Periode ist. So müßte für die Berechnung der Support-Werte im obigen Beispiel ermittelt werden, wie viele Sitzungen es insgesamt in einem Monat gab.

### 4.2.2 Relationale Transformation

Zur Speicherung der Muster in einer relationalen Datenbank wurde das temporale Regelmodell zunächst in das in Abbildung 4.1 dargestellte Entity-Relationship-Modell (ERM) transformiert. In jeder Periode wird eine Menge

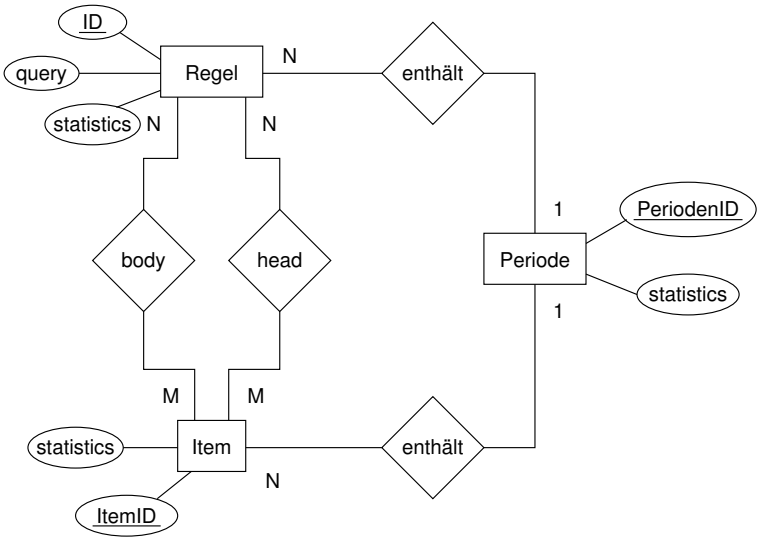


Abbildung 4.1: Entity-Relationship-Modell des temporalen Regelmodells

von Mustern entdeckt, die aus *Body* und *Head* bestehen. Da es auch Regeln gibt, die mehr als ein Element (*Item*) in *Body* und *Head* aufweisen, und ein Item in verschiedenen Regeln auf unterschiedliche Weise enthalten sein kann, wurden *Body* und *Head* als $N:M$-Beziehungen zwischen Item und Regel modelliert. Da die statistischen Eigenschaften eines Items ebenfalls nur für eine bestimmte Periode gelten, wurde – analog zur Modellierung einer Regel – eine zusätzliche Beziehung zwischen Item und Periode eingeführt. Um Items und Perioden referenzieren zu können, wurden für diese Entitäten zusätzlich Kennungen in das Modell aufgenommen.

Aus dem ERM wurde dann das Relationenschema, das in Abbildung 4.2 dargestellt ist, abgeleitet. Für alle Entitäten des ERM wurde eine separate

```
Regel (ID, PeriodenID, query, statistics)
Item (ItemID, PeriodenID, statistics)
Periode (PeriodenID, statistics)
body (ID, ItemID, PeriodenID)
head (ID, ItemID, PeriodenID)
```

Abbildung 4.2: Relationenschema des temporalen Regelmodells

Relation definiert. Zusätzlich wurden für die beiden $N:M$-Beziehungen Relationen definiert, welche die Schlüssel der jeweils beteiligten Entitäten als Fremdschlüssel speichern. Die Attribute, die den Primärschlüssel einer Relation bilden, sind unterstrichen. Das Attribut `PeriodenID` in den Relationen Regel und Item ist ein Fremdschlüssel, über den die Periode referenziert wird, zu der die Regel bzw. das Item jeweils gehört. Zu beachten ist, daß dieses Modell noch immer abstrakt ist, da `statistics` in den verschiedenen Relationen jeweils für eine Menge von statistischen Eigenschaften steht und in Abhängigkeit vom Anwendungsfall durch die tatsächlich betrachteten Eigenschaften ersetzt werden muß.

**Beispiel 4.2** Auf den über einen Zeitraum von mehreren Perioden gesammelten Transaktionsdaten eines Supermarktes ist eine Warenkorbanalyse durchgeführt worden. Für jede Transaktion wurde dabei aufgezeichnet, welche Produkte sie enthielt. Ziel der Analyse war es, für die verschiedenen Perioden zu bestimmen, welche Produkte von den Kunden gemeinsam gekauft wurden. Für jede Periode wurde erfaßt, wie viele Transaktionen es insgesamt gab, und für jedes Produkt wurde bestimmt, in wie vielen Transaktionen es ge-

kauft wurde. Die Güte der entdeckten Muster wurde mit Hilfe der Kriterien Support, Konfidenz und Lift gemessen (vgl. Abschnitt 2.2).

Zur Speicherung der Ergebnisse der Analyse wurde folgendes Relationenschema verwendet: Der wesentliche Unterschied zum allgemeinen Modell ist,

```
association (ID, queryID, pnum, support, confidence,
lift)
product (ID, pnum, counts, support)
period (pnum, counts)
body (associationID, productID, pnum, query)
head (associationID, productID, pnum, query)
query (queryID, query)
```

Abbildung 4.3: Relationenschema für eine Warenkorbanalyse

daß anstelle eines Attributs für alle Statistiken zusätzliche Spalten für die tatsächlich zu speichernden statistischen Eigenschaften aufgenommen wurden. So enthält das Attribut `counts` in der Relation `period` die Anzahl der Transaktionen in der jeweiligen Periode, in der Relation `product` die Anzahl der Transaktionen, die das jeweilige Produkt enthielten, und in der Relation `association` die Anzahl der Transaktionen, die die jeweilige Regel unterstützten. Zusätzlich wurde eine Relation `query` zur Speicherung der Mining-Anfragen verwendet, wodurch sich die Primärschlüssel der Relationen `association`, `body` und `head` ändern. ◇

### 4.2.3 Einbeziehung verschiedener Mining-Paradigmen

Die Form der Mining-Ergebnisse hängt natürlich vom verwendeten Mining-Paradigma ab (vgl. Abschnitt 2.2), weswegen nicht immer nur Assoziationsregeln zu verwalten sind. So bestehen die Ergebnisse eines Sequenz-Miners aus häufigen Sequenzen, während eine Cluster-Analyse Gruppen ähnlicher Objekte liefert. Obwohl das temporale Regelmodell ursprünglich nur für die Speicherung von Assoziationsregeln entworfen wurde, läßt es sich auch für die Verwaltung anderer Mustertypen verwenden.

#### Häufige Sequenzen

Für die Speicherung häufiger Sequenzen wird die gesamte Sequenz mit Ausnahme des letzten Elements als linke Seite (*Body*) einer Assoziation betrach-

tet, während das letzte Element die rechte Seite (*Head*) bildet. So würde das sequenzielle Muster $ABCDE$ als Assoziationsregel der Form $ABCD \Rightarrow E$ dargestellt, die ohne weitere Änderung unter Verwendung des vorgestellen Regelmodells gespeichert werden könnte. Die statistischen Eigenschaften der einzelnen Elemente des Musters werden dabei wieder zusammen mit den jeweiligen Elementen als strukturierte Objekte erfaßt.

Bei dieser Vorgehensweise ließen sich aber auch komplexere Sequenzen, z. B. Navigationsmuster, oder Entscheidungsbäume speichern. In einem solchen Fall besteht ein Muster aus einem Baum, der lediglich in alle in ihm enthaltenen Pfade zerlegt werden muß. Jeder Pfad enthält dann eine Sequenz von Knoten, die von der Wurzel des Baumes zu einem seiner Blätter führen und in Form häufiger Sequenzen gespeichert werden könnten. Demnach bestünde ein Muster $P$ aus einer *Menge* von Assoziationsregeln, d. h.

$$P = ((ID, query, body[\,], head[\,]), \{(timestamp, statistics)\}).$$

**Cluster**

Für die Speicherung der Ergebnisse einer Cluster-Analyse wird eine ähnliche Methode verwendet. In diesem Fall bestehen die Ergebnisse aus einer Menge von Gruppen ähnlicher Objekte. Jede dieser Gruppen kann durch ihren Mittelpunkt eindeutig identifiziert werden. Für jeden Cluster wird dann eine Assoziationsregel der Form $centroid \Rightarrow cluster$ gespeichert, wobei jedes Objekt in der Datenbank dem Cluster zugeordnet wird, zu dessen Mittelpunkt es die geringste Distanz aufweist. Natürlich unterscheiden sich die statistischen Eigenschaften, die für die Ergebnisse erfaßt werden, von denen einer Warenkorbanalyse. Diese Komponente des temporalen Regelmodells wurde jedoch bewußt generisch gestaltet, um die Eignung des Modells auch für andere Typen von Mining-Ergebnissen nicht einzuschränken.

## 4.3 Pflege des entdeckten Wissens

Für die Pflege der entdeckten Muster kommen in Abhängigkeit vom betrachteten Anwendungsfall verschiedene Vorgehensweisen in Frage. Besteht das primäre Interesse der Analyse in der Frage, welche Muster in der *Gesamtheit* der Daten enthalten sind, ist prinzipiell ein inkrementeller Ansatz angebracht. Sind dagegen die Änderungen, denen die Regelbasis im Zeitverlauf unterliegt, von größerer Wichtigkeit, ist diese Methode nicht geeignet. In diesem Fall ist

die Aktualisierung der Muster in Form der *Fortschreibung* der Muster notwendig, da sonst wichtige Entwicklungen in der Regelbasis verlorengehen können. Das folgende Beispiel soll die Auswirkungen der Entscheidung, wie die Analyse vorzunehmen ist, verdeutlichen.

**Beispiel 4.3** Ein Supermarkt verzeichnet wöchentlich 5000 Transaktionen. In drei aufeinanderfolgenden Wochen wird in jeweils 1000 Transaktionen das Produkt $A$ und in 1500 Transaktionen das Produkt $B$ verkauft. Während in den ersten beiden Wochen in 900 Transaktionen $A$ und $B$ gemeinsam verkauft wurden, waren es in der dritten Woche nur noch 600 Transaktionen.

Auf den Transaktionsdaten wird eine Warenkorbanalyse durchgeführt, wobei eine minimale Konfidenz von $\tau_c = 80\%$ und ein minimaler Support von $\tau_s = 10\%$ verwendet wird. Tabelle 4.1 faßt die Daten, die der Analyse zugrunde liegen, zusammen. Aus den Spalten $t_1$, $t_2$ und $t_3$ sind die Ergebnisse

| | $t_1$ | $t_2$ | $t_3$ | $t_{1-2}$ | $t_{1-3}$ |
|---|---|---|---|---|---|
| $\|D\|$ | 5000 | 5000 | 5000 | 10 000 | 15 000 |
| $\|A\|$ | 1000 | 1000 | 1000 | 2000 | 3000 |
| $s(A)$ | 20% | 20% | 20% | 20% | 20% |
| $\|B\|$ | 1500 | 1500 | 1500 | 3000 | 4500 |
| $s(B)$ | 30% | 30% | 30% | 30% | 30% |
| $\|A \cup B\|$ | 900 | 900 | 600 | 1800 | 2400 |
| $s(A \cup B)$ | 18% | 18% | 12% | 18% | 16% |
| $c(A \Rightarrow B)$ | 90% | 90% | 60% | 90% | 80% |
| $c(B \Rightarrow A)$ | 60% | 60% | 40% | 60% | 53,3% |

Tabelle 4.1: Ergebnisse der Warenkorbanalyse

der separaten Analyse der drei Perioden ersichtlich, während die Spalte $t_{1-2}$ die Ergebnisse des inkrementellen Ansatzes für die erste und zweite Periode und die Spalte $t_{1-3}$ die Ergebnisse der inkrementellen Analyse für die Perioden 1 bis 3 zeigt. Das Muster $B \Rightarrow A$ erreicht in keiner Periode die in der Mining-Anfrage vorgegebene Grenze für die Konfidenz von 80%. Das Muster $A \Rightarrow B$ hingegen weist nur in der dritten Periode eine Konfidenz auf, die kleiner als 80% ist. Dies wird jedoch nur offenbar, wenn alle Perioden separat analysiert werden. Wird hingegen der inkrementelle Ansatz verfolgt (vgl. Abschnitt 3.1), geht diese möglicherweise wichtige Tatsache verloren. Es wird

lediglich festgestellt, daß die Konfidenz auf 80% fällt, womit die vorgegebene Grenze jedoch zumindest erreicht wird. ◇

### 4.3.1 Inkrementelle Aktualisierung der entdeckten Muster

Bei der inkrementellen Aktualisierung der entdeckten Muster wird zu bestimmten Zeitpunkten $t_i$, $i \geq 1$ eine Analyse des Datensatzes $D_i$ durchgeführt, wobei $t_0$ der Zeitpunkt der ersten Analyse ist und $D_0$ den bis zum Zeitpunkt $t_0$ gesammelten Daten entspricht. Der im Zeitpunkt $t_i$ untersuchte Datensatz enthält alle Daten, die bisher gesammelt wurden, d. h. $D_{i-1} \cup D_i$. Jede Sitzung liefert eine Menge von Mustern, und für ein Muster $P_i$, das in der Mining-Session zum Zeitpunkt $t_i$ entdeckt wurde, werden die statistischen Eigenschaften in der Regelbasis mit dem Zeitstempel $t_i$ eingetragen. Ist das Muster $P_i$ dagegen zum ersten Mal gefunden worden, wird es in die Regelbasis aufgenommen.

Im Gegensatz zu den inkrementellen Mining-Techniken, die im Abschnitt 3.1 vorgestellt wurden, resultieren aus dieser Vorgehensweise zwar auch Zeitreihen für die statistischen Eigenschaften der Muster, es wird jedoch deutlich, daß diese die jeweils über den gesamten bisherigen Analysezeitraum *aggregierten* Werte enthalten. Auf der einen Seite liefert dieser Ansatz Ergebnisse, die weniger empfindlich auf kurzfristige Änderungen reagieren, andererseits kann es passieren, daß wichtige kurzfristige Änderungen übersehen und neu auftretende Muster nicht sofort erkannt werden – eine Problematik, die sich mit zunehmender Projektlänge verstärkt.

### 4.3.2 Fortschreibung der entdeckten Muster

Bei der Aktualisierung der entdeckten Muster durch Fortschreibung wird zu bestimmten Zeitpunkten $t_i$, $i \geq 1$ eine Analyse des Datensatzes $D_i$ durchgeführt, wobei $t_0$ der Zeitpunkt der ersten Analyse ist und $D_0$ den bis zum Zeitpunkt $t_0$ gesammelten Daten entspricht. Der im Zeitpunkt $t_i$ untersuchte Datensatz enthält alle Daten, die im Zeitraum $t_i - t_{i-1}$ gesammelt wurden. Bei der Eintragung der Muster entspricht die Vorgehensweise der des inkrementellen Ansatzes.

Die Aktualität der Ergebnisse ist bei Fortschreibung der Muster natürlich höher als bei der inkrementellen Aktualisierung. Selbst bei Analysen über lange Zeiträume hinweg werden neue Muster entdeckt, sowie sie in den Daten

einer einzelnen Periode ausreichend stark vertreten sind. Auch die Zeitreihen der statistischen Eigenschaften sind aussagekräftiger, da sie keine aggregierten Werte enthalten. Obwohl oft im Interesse des Anwenders, reagiert diese Methode vergleichsweise stark auf kurzfristige Änderungen in den Daten. Dadurch sinkt jedoch auch die Gefahr, wichtige Änderungen zu übersehen.

### Aktualisierung von Assoziationsregeln

Für die Aktualisierung von Assoziationsregeln kommt jedoch auch eine Technik in Frage, die fast vollständig ohne Miner auskommt [Baron u. a. 2003]. Dabei werden in der Datenbank nicht nur die gefundenen Muster gespeichert, sondern auch die Transaktionsdaten selbst. Zum Zeitpunkt $t_0$ wird eine Mining-Session durchgeführt, um eine initiale Menge von Mustern zu bestimmen. In den Folgeperioden können dann für bekannte Muster die Zeitreihen der statistischen Eigenschaften fortgeschrieben werden, indem die Werte direkt aus den Transaktionsdaten bestimmt werden. Zu diesem Zweck werden auf Basis des Musters drei SQL-Anfragen formuliert: eine Anfrage, die die Anzahl der Transaktionen in der betrachteten Periode bestimmt; eine weitere Anfrage, die die Anzahl der Transaktionen bestimmt, die die Regel unterstützen; eine dritte Anfrage, welche die Anzahl der Transaktionen bestimmt, die die linke Seite (*Body*) enthalten. Aus den erhaltenen Werten lassen sich dann Support und Konfidenz des Musters berechnen, aber auch andere Maße wie Lift und *Certainty Factor* (vgl. Abschnitte 2.2 und 5.3.1). Wird der Support der rechten Seite (*Head*) des Musters nicht im Rahmen der Ermittlung der statistischen Eigenschaften eines anderen Musters bestimmt, müßte eine weitere SQL-Anfrage formuliert werden, die die Anzahl der Transaktionen, die die rechte Seite des Musters enthalten, bestimmt.

Die Komplexität der Anfragen steigt dabei mit der Länge der zu aktualisierenden Muster, wobei die Verarbeitungsdauer der Anfragen natürlich auch von der Größe des Transaktionsprotokolls und den Optimierungen, die im physischen Datenbank-Design vorgenommen wurden, abhängt. Nimmt man an, daß es sich teilweise um ähnliche Muster handelt, die inhaltliche Überlappungen aufweisen, müssen jedoch nicht immer alle Anfragen gestellt werden. Die Anfrage, die die Gesamtzahl der Transaktionen in einer Periode bestimmt, braucht beispielsweise nur einmal bearbeitet zu werden. Der Aufwand, den die gezielten Anfragen verursachen, sollte jedoch in den meisten Fällen wesentlich kleiner sein als im Fall der ungerichteten Suche nach *allen* Mustern einer Mining-Software. Wesentlicher Nachteil dieser Methode ist je-

doch, daß in den Zeitpunkten $t_i$, $i \geq 1$ keine neuen Muster gefunden werden können, da die Anfragen auf Basis der bekannten Muster erstellt werden. Insofern wäre zumindest die periodische Aktualisierung der gesamten Regelbasis erforderlich (vgl. Abschnitt 6.3.2).

**Beispiel 4.4** In der Mining-Session zum Zeitpunkt $t_0$ sei die Assoziationsregel $AB \Rightarrow C$ entdeckt worden, die zum Zeitpunkt $t_1$ aktualisiert werden soll. Angenommen die Tabelle `trans01` enthält den Datensatz $D_1$, der im Zeitraum $t_1 - t_0$ gesammelt wurde, dann können die in Abbildung 4.4 dargestellten SQL-Anfragen verwendet werden, um die Anzahl der Transaktionen in $t_1$, die Anzahl der Transaktionen in $t_1$, die $AB$ unterstützen, und die Anzahl der Transaktionen in $t_1$, die $ABC$ unterstützen, zu bestimmen. Hierbei

```
select count(distinct trans_id)
  from trans01;

select count(distinct trans_id)
  from trans01 t1, trans01 t2
  where t1.trans_id = t2.trans_id
  and t1.item = 'A' and t2.item = 'B';

select count(distinct trans_id)
  from trans01 t1, trans01 t2, trans01 t3
  where t1.trans_id = t2.trans_id
        and t2.trans_id = t3.transid
  and t1.item = 'A'
      and t2.item = 'B'
      and t3.item = 'C';
```

Abbildung 4.4: SQL-basierte Bestimmung statistischer Eigenschaften

ist `trans_id` eine Kennung, die eine Transaktion eindeutig identifiziert. Die Resultate der Anfragen können dann verwendet werden, um die Werte der statistischen Eigenschaften zu berechnen. ◇

### Aktualisierung von Cluster-Beschreibungen

Bei der Aktualisierung von Cluster-Beschreibungen stellt sich das Problem der eindeutigen Identifizierung eines Clusters, da sich selbst für zwei auf-

einanderfolgende Mining-Sessions nur äußerst selten zweifelsfrei bestimmen läßt, welchem Cluster aus der Vorperiode ein gegebener Cluster entspricht. Während sich die Kennung einer Assoziationsregel aus ihrem Inhalt ergibt, erweist sich diese Art der Ermittlung aufgrund der Vielzahl von Objekten, die zu einem Cluster gehören können, als ungeeignet. Statt dessen wird für jeden Cluster zum Zeitpunkt $t_0$ ein abstraktes Kennzeichen (*Label*) erzeugt, das dann in den folgenden Perioden verwendet wird. Für die Fortschreibung kommt ein der Aktualisierung von Assoziationsregeln vergleichbarer Ansatz zur Anwendung. Zum Zeitpunkt $t_i$ wird jedes Objekt des Datensatzes $D_i$, der im Zeitraum $t_i - t_{i-1}$ gesammelt wurde, einem der Cluster, die in einer Periode $t_j$, $0 \leq j < i$ gefunden wurden, zugeordnet.[2] Nach der Zuordnung werden die statistischen Eigenschaften der Cluster neu berechnet und unter Verwendung des in $t_0$ festgelegten Labels in die Datenbank eingetragen.

Auch in diesem Fall bleibt jedoch die Problematik der Aktualität der fortgeschriebenen Ergebnisse. Mit wachsender Länge des Analysezeitraums steigt die Wahrscheinlichkeit, daß die Cluster-Beschreibungen die Population nicht mehr hinreichend repräsentieren. Deshalb wurden verschiedene Kriterien abgeleitet, die auf den statistischen Eigenschaften der Cluster und den Cluster-Beschreibungen beruhen und die Abschätzung der *Gültigkeit* der bestehenden Cluster-Beschreibungen ermöglichen.

Abbildung 4.5 zeigt das idealisierte Ergebnis einer Cluster-Analyse. Aus den Datenpunkten wurden drei Cluster $C_1$, $C_2$ und $C_3$ abgeleitet, für die entsprechend die Mittelpunkte $m_1$, $m_2$ und $m_3$ angegeben sind. Für jeweils zwei Cluster $C_k$ und $C_l$ ist eine Grenze $g_{kl}$ angegeben, nach der entschieden wurde, zu welchem Cluster ein gegebenes Objekt gehört. Der um den Mittelpunkt eines Clusters gezeichnete Kreis stellt den *Kern* des Clusters $K(C_k)$ dar. Sein Radius entspricht dem mittleren quadrierten Abstand zwischen den Objekten, die zu diesem Cluster gehören, und dem Centroid. Weiterhin sei $s(C_k)$ der Support eines Clusters, der sich aus der Anzahl der Objekte, die zu Cluster $C_k$ gehören, bezogen auf die Größe der Population $|D_i|$, ergibt.

Neben einfachen Maßen wie der absoluten und/oder relativen Abweichung der statistischen Maßzahlen kommt eine Verwendung der Grenzen $g_{kl}$ in Verbindung mit den Kernen der Cluster $K(C_k)$ in Frage, um die Aktualität des Clusterings einzuschätzen. Zu jedem Zeitpunkt $t_i$ werden die Objekte aus $D_i$ den in einer der Vorperioden $t_j$ gefundenen bzw. aktualisierten Gruppen

[2]Diese Methode entspricht im wesentlichen der Klassifikation, wobei die Klassen ursprünglich nicht vorgegeben waren, sondern in einer Vorperiode ermittelt wurden.

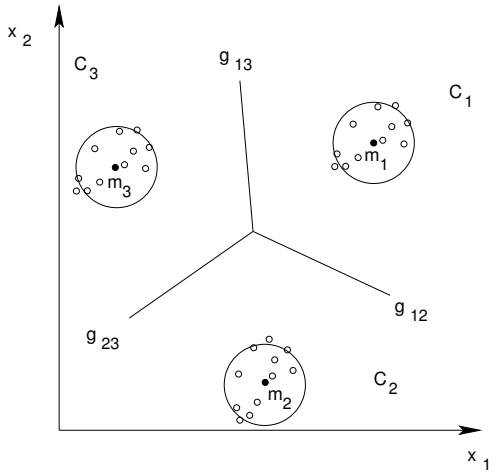


Abbildung 4.5: Idealisiertes Ergebnis einer Cluster-Analyse

zugeordnet. Nach der Berechnung der Statistiken wird überprüft, in welchem Maße sich $g_{kl}$ und $K(C_k)$ geändert haben (vgl. Abbildung 4.6).

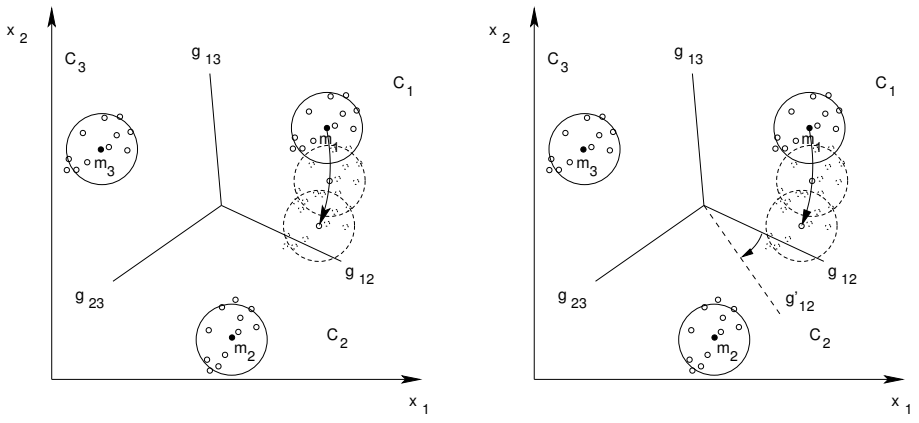


Abbildung 4.6: Gültigkeit von Cluster-Beschreibungen

In Abhängigkeit von der gewünschten Aktualität sind drei unterschiedliche Methoden zur Bestimmung der Gültigkeit der Cluster-Beschreibungen anwendbar:

1. Statische Prüfung

   Bei der statischen Prüfung werden in jeder Periode $t_i$ die Cluster-Beschreibungen, die in Periode $t_0$ ermittelt wurden, zur Klassifikation verwendet. Anschließend werden die Cluster-Beschreibungen – insbesondere Mittelpunkt und Standardabweichung – aktualisiert und in die Datenbank eingetragen. Das Clustering wird ungültig, wenn der Kern eines Clusters $K(C_k)$ eine der ursprünglichen Grenzen $g_{kl}$ schneidet. In diesem Fall würde dem Nutzer die Notwendigkeit einer neuen Mining-Session signalisiert.

2. Dynamische Prüfung

   Bei dieser Methode werden für die Klassifikation der Daten in $t_i$ die in der Vorperiode $t_{i-1}$ aktualisierten Cluster verwendet, für die Bestimmung der Gültigkeit des Clusterings werden hingegen weiterhin die Grenzen aus der Periode $t_0$ verwendet (vgl. linker Teil von Abbildung 4.6).

3. Reine Fortschreibung

   In diesem Fall werden sowohl die Cluster-Beschreibungen als auch die Grenzen $g_{kl}$ in jeder Periode aktualisiert. Das bedeutet, daß auch die Gültigkeit des Clusterings anhand der Werte der Vorperiode überprüft wird (vgl. rechter Teil von Abbildung 4.6).

Es zeigt sich, daß die Ungültigkeit bei Verwendung des letztgenannten Ansatzes unwahrscheinlicher wird, da auch die Grenzen der Cluster von Periode zu Periode angepaßt werden. Nur bei extremen Änderungen der Population zwischen zwei aufeinanderfolgenden Perioden können die Cluster-Beschreibungen ungültig werden. Deshalb sollten bei Verwendung dieses Ansatzes auch die statistischen Eigenschaften der Cluster überwacht werden, z. B. unter Verwendung eines Differenz-Parameters, der zur Erkennung relativer Änderungen bestimmter Stärke verwendet wird [Baron und Spiliopoulou 2001].[3] Steht die Gültigkeit der Cluster im Vordergrund der Analyse, sollte in jedem Fall einer der anderen Ansätze verfolgt werden.

[3] Diese Kombination ist natürlich auch für die anderen Herangehensweisen möglich.

## 4.4 Zusammenfassung

Sollen zeitliche Gültigkeit und Änderungen von Mustern erkannt werden, ist die Beachtung ihrer temporalen Dimension notwendig. Um die Auswertung dieses Aspekts zu automatisieren, ist zudem eine adäquate Verwaltung der gefundenen Regeln nötig. In diesem Kapitel wurde ein Datenmodell vorgestellt, das als Grundlage für die Speicherung des entdeckten Wissens dienen kann. Im Unterschied zu anderen Arbeiten wird hier ein Muster als ein komplexes Objekt angesehen, dessen verschiedene Bestandteile als *Einheit* betrachtet werden. Demnach ist eine Regel ein temporales Objekt, das aus einem invarianten Teil und einer dynamischen Komponente besteht. Der statische Teil widerspiegelt jene Eigenschaften des Musters, die sich im Zeitverlauf nicht ändern. Dazu zählt der Inhalt des Musters, der den gefundenen Zusammenhang in den Daten beschreibt, und eine Regelkennung, die ein Muster über alle Perioden eindeutig identifiziert. Zusätzlich wird die Mining-Anfrage gespeichert, welche die statistischen Kriterien für die Musterentdeckung festlegte. Der variable Teil des Musters besteht aus den statistischen Eigenschaften, die zusammen mit dem Zeitstempel der Mining-Session gespeichert werden. Unter Verwendung dieses Modells können alle Aspekte der entdeckten Muster erfaßt werden und stehen somit für weitere Analysen zur Verfügung.

Für die Verwaltung der Regeln in einer Datenbank wurde das temporale Regelmodell der im Abschnitt 4.2.2 beschriebenen relationalen Transformation unterzogen. Zu diesem Zweck wurde das Objektmodell zunächst in das in Abbildung 4.1 dargestellte ER-Modell überführt, aus dem anschließend ein Relationenschema abgeleitet wurde. Der Teil des Modells, der die statistischen Eigenschaften abbildet, wurde hierbei bewußt generisch gestaltet und kann um spezifische Attribute erweitert werden. Auch die Einbeziehung der Ergebnisse verschiedener Mining-Paradigmen wie häufige Sequenzen, Cluster-Beschreibungen, Navigationsmuster und Entscheidungsbäume läßt sich unter Verwendung des temporalen Regelmodells problemlos realisieren.

Für die Pflege des entdeckten Wissens bieten sich in Abhängigkeit vom betrachteten Anwendungsfall unterschiedliche Optionen. Liegt der Schwerpunkt der Analyse in der Fragestellung, *welche* Muster über den gesamten Analysezeitraum in den Daten enthalten sind, kann ein inkrementeller Ansatz verfolgt werden. In diesem Fall werden alle Daten, die bis zum Zeitpunkt der jeweiligen Mining-Session gesammelt wurden, analysiert. Die Alternative liegt in einer reinen Fortschreibung der Muster, die durch eine Analyse

der Daten, die zwischen der letzten Mining-Session und der aktuellen gesammelt wurden, vorgenommen wird. Bei diesem Ansatz wird die Aktualität der Ergebnisse stärker betont, da jede Sitzung nur jene Muster liefert, die in den Daten der zurückliegenden Periode enthalten sind. Beide Ansätze haben Vor- und Nachteile im Hinblick auf die Sensitivität und Aktualität, die im Abschnitt 4.3 diskutiert wurden.

Zur Fortschreibung von Assoziationsregeln wurde eine SQL-basierte Methode vorgeschlagen, bei der die statistischen Eigenschaften einer gegebenen Menge von Mustern direkt aus den zugrunde liegenden Transaktionsdaten bestimmt werden. Das Problem der eindeutigen Identifizierung eines Clusters verhindert die Anwendung dieses Verfahrens zur Pflege eines Clusterings. Für diesen Fall kommt eine zweigeteilte Strategie in Betracht: Die in einer Sitzung gefundenen Cluster-Beschreibungen werden in den Folgeperioden zur Klassifikation neuer Objekte verwendet. In diesem Zusammenhang ergibt sich jedoch das Problem der Aktualität des verwendeten Clusterings, wobei sich die unterbreiteten Lösungsvorschläge hinsichtlich der Toleranz gegenüber den Änderungen in der Population unterscheiden.

# Kapitel 5

# Überwachung des entdeckten Wissens

## 5.1 Einführung

Grundlage der Überwachung von Mustern ist das temporale Regelmodell, das im vorigen Kapitel vorgestellt wurde. Erst durch die Verwendung eines Datenmodells, das die zeitliche Dimension der Muster einbezieht, wird die Erkennung von Änderungen und damit ihre Überwachung möglich. In diesem Kapitel soll es neben der Erkennung vor allem um die *Bewertung* von Musteränderungen gehen. Während es bei der Erkennung von Änderungen um die verschiedenen Ebenen geht, auf denen Änderungen verfolgt werden können, widmet sich die Bewertung von Änderungen den Kriterien, nach denen Musteränderungen sinnvoll beurteilt werden können.

Oft liefert schon die Analyse von Datensätzen mit moderater Größe eine Vielzahl von Mustern, deren manuelle Inspektion nahezu unmöglich ist. Wird dann der im vorigen Kapitel vorgestellte Ansatz zur Aktualisierung der entdeckten Muster verwendet, fallen in jeder Periode Musteränderungen in vergleichbarem Umfang an. Das ursprüngliche Problem, welche Muster dem Nutzer angezeigt werden sollten, gewinnt bei den Musteränderungen an Brisanz. Aus diesem Grund ist es notwendig, Kriterien zur Bewertung von Änderungen abzuleiten, die dem Nutzer die Abschätzung ihrer *Wichtigkeit* ermöglichen. Dabei stellt sich außerdem die Frage, ob konventionelle Maßstäbe zur Bewertung der Wichtigkeit von Mustern auch für die Beurteilung der Musteränderungen herangezogen werden können. Andererseits muß

geklärt werden, ob nicht auch temporale Faktoren verwendet werden können, um die konventionellen Kriterien zur Bewertung der Muster zu ergänzen.

Während die zeitliche Dimension zwingend notwendig ist, um Musteränderungen erkennen zu können, weisen die Änderungen selbst natürlich auch temporale Eigenschaften auf. So argumentieren Chakrabarti u. a. [1998], daß sich der größte Nutzen für den Anwender aus der Erkennung von Änderungen ergibt, wenn er mit den in den Daten entdeckten Modellen erst einmal vertraut ist. Folgt man dieser Argumentation konsequent, gilt die nächste Überlegung natürlich der zeitlichen Dimension der Musteränderungen. Die Beantwortung der Frage, ob eine Änderung kurzfristiger Natur ist oder einen langfristigen Trend widerspiegelt, liefert zusätzliche Informationen über ein Muster und kann die Entscheidung, welche Musteränderungen für den Anwender von Interesse sein könnten, maßgeblich beeinflussen.

Der letzte Teil des Kapitels widmet sich der Frage, *wodurch* die Änderungen der Regelbasis hervorgerufen wurden. Natürlich sind nur in den seltensten Fällen alle Faktoren, die einen Datensatz beeinflussen können, selbst darin enthalten. Aber der Idee folgend, daß detailliertere Informationen über die in den Daten repräsentierten Modelle eine Identifizierung der realen Ereignisse, die zu den Änderungen geführt haben, ermöglichen, wird der Versuch unternommen, all jene Ursachen zu bestimmen, die sich direkt aus dem untersuchten Datensatz ergeben.

## 5.2 Erkennung von Musteränderungen

Durch die Nutzung eines temporalen Datenmodells ist das grundlegende Problem für die Erkennung eines Teils der möglichen Änderungen bereits gelöst. Jedes Muster, das in der Regelbasis gespeichert ist, weist für jede erfaßte statistische Eigenschaft eine Historie ihrer Werte auf. Änderungen dieser Eigenschaften lassen sich somit ihren Zeitreihen unmittelbar entnehmen. Von möglichen Änderungen sind jedoch nicht nur die statistischen Eigenschaften eines Musters betroffen, auch der Inhalt eines Musters kann sich ändern. Hierbei stellt sich insbesondere die Frage der *Semantik* einer solchen Änderung – per se ist nicht festzustellen, ob es sich *wirklich* um eine Änderung handelt. Das folgende Beispiel soll die Problematik verdeutlichen.

**Beispiel 5.1** In einer Mining-Session zum Zeitpunkt $t_i$ sei eine Assoziationsregel $AB \Rightarrow CD$ entdeckt worden. In einer Folgeperiode $t_j$, $j > i$ wird statt

dessen das Muster $AB \Rightarrow CE$ gefunden. Dabei ist zu fragen, ob sich der Inhalt des alten Musters geändert hat oder ob ein neues Muster gefunden wurde. Die Antwort hängt im wesentlichen von der Art der verwendeten Kennung ab. Gibt es eine *globale* Kennung, die jedes Muster über *alle* Perioden eindeutig identifiziert, so liefert ein Vergleich der Kennungen der beiden Regeln die Antwort. Stimmen die Kennungen überein, handelt es sich auch um die gleiche Regel und damit um eine Regeländerung, andernfalls um eine neue und damit andere Regel. ◇

Obwohl inhaltliche Änderungen im vorgestellten Konzept vorkommen und im folgenden auch ein Vorschlag zur Verfolgung inhaltlicher Änderungen gemacht wird, ist eine vollständige Lösung dieses Problems nicht Bestandteil dieser Arbeit.

### 5.2.1 Änderungen von Mustern

Für die Änderungen eines einzelnen Musters kommen prinzipiell beide Bestandteile des Musters in Betracht, seine statistischen Eigenschaften und sein Inhalt. Hinsichtlich der Änderungen der statistischen Eigenschaften einer Regel definieren wir die *Evolution* einer Regel wie folgt [Baron und Spiliopoulou 2001]:

**Definition 5.1** Als *Evolution* eines Musters werden alle Änderungen der statistischen Eigenschaften $(p_1, p_2, \ldots, p_n)$ des Musters im Zeitverlauf bezeichnet, welche die vorgegebenen Grenzen für diese Maßzahlen $(\tau_{p_1}, \tau_{p_2}, \ldots, \tau_{p_n})$ nicht verletzen.

Im Normalfall gibt der Nutzer in der Mining-Anfrage untere, seltener obere Schranken für die statistischen Eigenschaften des Musters wie Support und Konfidenz an, um die Ausprägungen der statistischen Eigenschaften jener Muster zu beschreiben, an denen er interessiert ist. Im Rahmen einer Mining-Sitzung werden dann all jene Regeln gefunden, die diese Anforderungen erfüllen. Änderungen der statistischen Eigenschaften eines Musters im Zeitverlauf werden gemäß Definition 5.1 nur so lange toleriert, wie sie nicht die *Gültigkeit* des Musters beeinflussen, d. h. nicht die angegebenen Schwellwerte unter- bzw. überschreiten.

Allerdings gilt diese Einschränkung nur, wenn eine konventionelle Mining-Software zur Entdeckung der Muster eingesetzt wird. In diesem Fall müßten

die Grenzen manuell angepaßt werden, um festzustellen, welche Änderungen dazu geführt haben, daß die Regel nicht mehr gefunden werden konnte. Das ist zwar möglich, im allgemeinen jedoch sehr aufwendig. Wird statt dessen eine Methode zur Aktualisierung von Assoziationsregeln verwendet, wie sie im Abschnitt 4.3.2 vorgestellt wurde, werden die statistischen Eigenschaften direkt aus den der Analyse zugrunde liegenden Transaktionsdaten bestimmt. Das bedeutet, daß auch Änderungen verfolgt werden können, welche die Grenzen der statistischen Eigenschaften verletzen. Unter Verwendung dieses Ansatzes kann die *Evolution* eines Musters weiter gefaßt werden:

**Definition 5.2** Als die *unbeschränkte Evolution* eines Musters werden alle Änderungen der statistischen Eigenschaften $(p_1, p_2, \ldots, p_n)$ des Musters im Zeitverlauf bezeichnet.

Dieser Punkt ist für die Bewertung von Musteränderungen von großem Interesse: Für den Nutzer ist es wichtig zu wissen, *warum* ein bestimmtes Muster nicht mehr in den Daten vertreten ist – handelt es sich um eine minimale Änderung, so daß die Grenze nur geringfügig überschritten wurde, oder handelt es sich um eine deutliche Änderung, die für grundlegende Änderungen in der Population steht. Voraussetzung ist jedoch, daß ein Mechanismus zur Verfügung steht, der die Ermittlung der Statistiken aller Muster ermöglicht.

Für die Änderungen eines Musters hinsichtlich seines Inhalts definieren wir die *Mutation* eines Musters wie folgt:

**Definition 5.3** Als *Mutation* einer Regel werden alle inhaltlichen Änderungen des Musters im Zeitverlauf bezeichnet, die nicht zu einer Änderung der Kennung für diese Regel führen.

Die Einschränkung in Definition 5.3 ist für die Beobachtbarkeit inhaltlicher Änderungen von wesentlicher Bedeutung. Wie im Beispiel 5.1 ausgeführt wurde, können inhaltliche Musteränderungen nur verfolgt werden, wenn eine Regel anhand einer globalen Kennung über alle Perioden hinweg eindeutig identifiziert werden kann. Anderenfalls ließe sich die Frage, ob es sich um eine Änderung oder ein neues Muster handelt, nicht beantworten. Für das temporale Regelmodell in der im Abschnitt 4.2.1 vorgestellten Form folgt aus Definition 5.3, daß sich inhaltliche Änderungen *nicht* verfolgen lassen. Die Kennung wird in diesem Fall aus dem gesamten Inhalt des Musters bestimmt und ändert sich deshalb bei jeder Änderung des Inhalts.

Aber auch diese Einschränkung läßt sich zumindest zum Teil aufheben. Es ist lediglich notwendig, die Vorgehensweise zur Ermittlung der Kennung eines Musters zu ändern. Im Augenblick wird eine Funktion verwendet, die den gesamten Inhalt der Regel berücksichtigt. Bezieht die Funktion hingegen beliebige Teile des Musters ein, würden alle Inhaltsänderungen, die diesen Teil nicht betreffen, nicht zur Änderung der Kennung führen und wären damit beobachtbar. So könnte beispielsweise nur die rechte Seite (*Head*) verwendet werden, was zur Folge hätte, daß alle Muster mit der gleichen rechten Seite die gleiche Kennung hätten. In einem solchen Fall würden nicht mehr einzelne Regeln beobachtet, sondern Regelgruppen. Dies kann in Abhängigkeit vom Anwendungskontext durchaus gewünscht sein, es muß jedoch beachtet werden, daß sich dadurch auch die beobachtbaren statistischen Eigenschaften ändern. So wäre die Fortschreibung von statistischen Eigenschaften einzelner Regeln praktisch unmöglich, während für eine Regelgruppe zusätzliche Eigenschaften erfaßt werden könnten, wie z. B. der Support einer Gruppe, d. h. der Anteil der Regeln in einer Gruppe, bezogen auf die Gesamtzahl der gefundenen Regeln.

Welcher Ansatz verfolgt wird, muß letztlich im Einklang mit den konkreten Zielen der Analyse entschieden werden. Dem Informationsgewinn, der aus den zusätzlichen Erkenntnissen durch die Erkennung von inhaltlichen Änderungen erwächst, steht der Informationsverlust durch die Aggregation der statistischen Eigenschaften der Muster gegenüber. Die vorliegende Arbeit enthält über den gemachten Vorschlag hinaus keine Lösung für diese Problematik. Bei der Implementierung wurden jedoch bereits die beiden offensichtlichen Möglichkeiten, die Kennung auf Grundlage der linken oder rechten Seite des Musters zu bestimmen, einbezogen, indem zwei zusätzliche Gruppen-Kennungen eingeführt wurden, die bei Bedarf verwendet werden können.

### 5.2.2 Änderungen der Regelbasis

Nachdem die Änderungen, die auf der Ebene eines einzelnen Musters zu erwarten sind, definiert wurden, soll kurz die Frage diskutiert werden, von welchen Änderungen die Regelbasis betroffen sein kann.

Von Interesse ist hier die Anzahl der Regeln, die zu einem gegebenen Zeitpunkt $t_i$ in der Regelbasis enthalten sind. Solange alle Muster nur beobachtbare Änderungen aufweisen, d. h. Änderungen gemäß den Definitionen 5.1 und 5.3, wird die Regelbasis keine Änderungen zeigen. Interessant sind je-

doch Fälle, die durch die Definition 5.2, nicht jedoch durch die Definition 5.1 abgedeckt sind, da sich in diesem Fall möglicherweise die Anzahl *sichtbarer* Regeln ändert. Für den Anwender ist dabei von besonderem Interesse, welche Muster seit der letzten Mining-Session hinzugekommen, welche Muster verschwunden sind sowie die Menge der Muster, die stabil in der Regelbasis vertreten sind.

Auf dieser Grundlage läßt sich die Regelbasis in Gruppen von Mustern einteilen, die die gleiche bzw. eine ähnliche Stabilität über die Zeit aufweisen [Baron und Spiliopoulou 2003]. Sei $f(P)$ die Frequenz, mit der ein Muster $P$ im Zeitverlauf erscheint, d. h. der Anteil der Mining-Sessions, in denen das Muster entdeckt wurde, und $[0, 1]$ der zulässige Bereich von $f$. Dann ließe sich $f$ in die Intervalle $I_L = [0, 0.5)$, $I_M := [0.5, 0.75)$, $I_H := [0.75, 0.9)$, $I_{H+} := [0.9, 1)$ und $I_{permanent} := [1, 1]$ einteilen, und jedes Muster der Regelbasis ließe sich zu jedem Zeitpunkt $t_i$ einer dieser Gruppen zuordnen, wobei für große Werte von $i$ das Intervall $I_{permanent}$ alternativ auch auf $[0.9, 1]$ gesetzt werden kann. Muster, deren Frequenz in die Intervalle $I_H$ bzw. $I_{H+}$ fällt, werden dann als *häufig* bezeichnet, und Muster, deren Frequenz in die Intervalle $I_L$ und $I_M$ fällt, werden als *temporär* bezeichnet. Im Gegensatz dazu werden Muster, die in jeder Periode gefunden werden, zur Gruppe der *permanenten* Muster gezählt. Die Entwicklung dieser Gruppen im Zeitverlauf ließe dann Rückschlüsse auf die Stabilität der im Datensatz gefundenen Modelle zu. Weist die Gesamtzahl der Muster starke Änderungen auf oder treten in den Gruppen starke Fluktuationen auf, dann sind die in den Daten gefundenen Zusammenhänge weniger stabil und sollten häufiger überprüft werden. Auf diese Weise ließe sich z. B. ermitteln, ob der Zeitraum zwischen zwei aufeinanderfolgenden Mining-Sessions oder die Zeitpunkte selbst eventuell falsch gewählt sind und geändert werden sollten.

## 5.3 Bewertung von Musteränderungen

Wie in der Einleitung zu diesem Kapitel bereits erwähnt wurde, ist im Normalfall der Umfang der Ergebnisse einer einzelnen Mining-Session so groß, daß die Ergebnisse kaum noch manuell gesichtet werden können. Ist die Regelbasis selbst keinen größeren Änderungen ausgesetzt, kommen in jeder Periode Musteränderungen in vergleichbarer Anzahl dazu. Aus diesem Grund ist es unerläßlich, die Menge der gemeldeten Änderungen zu filtern und dem Nutzer möglichst nur noch jene Änderungen zu melden, an denen er in-

teressiert ist. Das *Interesse* eines Nutzers ist jedoch ein äußerst subjektives Konzept, weswegen die Frage, welche Änderungen anzuzeigen sind, nur sehr schwer beantwortet werden kann. In diesem Abschnitt soll zum einen versucht werden, konventionelle Maßstäbe zur Abschätzung der Wichtigkeit einer Musteränderung heranzuziehen. Zusätzlich wird zwischen subjektiven Kriterien, d. h. Kriterien, die sich aus dem Anwendungskontext ergeben, und objektiven Kriterien, die unabhängig von der spezifischen Anwendung gelten, unterschieden. Bei den objektiven Kriterien sei insbesondere das Konzept der statistischen Signifikanz hervorgehoben, das zwischen zufälligen Schwankungen und echten Schwankungen, die *nicht* durch das statistische Konzept des Zufalls erklärt werden können, unterscheidet, indem ein Bezug zur Grundgesamtheit hergestellt wird. Zum anderen werden verschiedene Heuristiken entwickelt, die Musteränderungen nach mehr als nur einem Kriterium bewerten können.

### 5.3.1 Wichtigkeit von Mustern

Ist das Kriterium zur Auswahl der interessanten Änderungen die Wichtigkeit, die der Anwendungsexperte den Mustern beimißt, stellt sich das Problem der Bewertung von Musteränderungen eigentlich nicht mehr, da die Auswahl bereits vor Beginn der eigentlichen Überwachung erfolgte. Für jedes Muster, das in Periode $t_i$ gefunden wurde, wird überprüft, ob es die Kriterien des Anwenders hinsichtlich der Wichtigkeit erfüllt. Änderungen werden dann nur für jene Muster gemeldet, die die festgelegten Kriterien erfüllen.

Obwohl dieses Prinzip sehr einfach ist, löst es nicht das eigentliche Problem. Änderungen werden zwar nur noch für interessante Muster gemeldet, aber es stellt sich weiterhin die Frage, ob die Änderung an sich auch interessant ist. Es besteht die Gefahr, daß dem Nutzer eine Reihe von Änderungen gemeldet werden, bei denen sich die Werte der statistischen Eigenschaften z. B. nur unwesentlich von denen der Vorperiode unterscheiden. Auf der anderen Seite können sehr starke Änderungen übersehen werden, weil das entsprechende Muster die Kriterien, durch die die Wichtigkeit der Muster festgelegt wird, nicht erfüllt.

**Beispiel 5.2** Ein Muster $A \Rightarrow B$ weise in Periode $t_0$ einen Support von $s = 5\%$ und eine Konfidenz von $c = 10\%$ auf und in Periode $t_1$ einen Support von $s = 30\%$ und eine Konfidenz von $c = 75\%$. Der Anwender habe die untere Grenze für den Support auf $\tau_s = 20\%$ und für die Konfidenz auf

$\tau_c = 80\%$ festgelegt. Obwohl das Muster sehr starke Änderungen für den Support *und* die Konfidenz aufweist – der Support liegt dadurch sogar über dem kritischen Wert –, wird die Änderung nicht angezeigt, da die Konfidenz des Musters auch in Periode $t_1$ unter der vom Nutzer angegebenen Grenze liegt. ◇

Dies ist natürlich ein extremes Beispiel, aber es verdeutlicht sehr gut die allgemeine Problematik bei der Bewertung von Musteränderungen. Die Methoden zur Bewertung der Wichtigkeit eines Musters basieren zumeist auf den statischen Eigenschaften der Muster. Soll dagegen ihre Dynamik eingeschätzt werden, d. h. die Änderungen, die sie im Zeitverlauf zeigen, sind diese Methoden deutlich schlechter geeignet.

### 5.3.2 Subjektive Ermittlung des Interesses

Das subjektive Interesse des Anwenders an Regeländerungen ergibt sich prinzipiell aus dem Anwendungskontext. Ähnlich wie im vorhergehenden Abschnitt legt der Anwender die Kriterien fest, nach denen die interessanten Muster ausgewählt werden sollen. Im Unterschied zur im Abschnitt 5.3.1 vorgestellten Vorgehensweise ergeben sich diese Kriterien jedoch aus den Zielen der Analyse. Für die statistischen Eigenschaften dieser Muster gibt der Anwender dann untere und/oder obere Grenzen an, die nicht über- bzw. unterschritten werden dürfen. Statt also festzulegen, welche statischen Kriterien die Werte der beobachteten Statistiken erfüllen müssen, bestimmt er, in welchen Grenzen sie sich ändern dürfen. Nach diesem Ansatz sind nur solche Musteränderungen von Belang, die (a) ein interessantes Muster betreffen und (b) zur Verletzung einer bestimmten, vorher festgelegten Grenze führen. Beispiel 5.3 soll den Unterschied zur Methode im Abschnitt 5.3.1 verdeutlichen.

**Beispiel 5.3** Der Betreiber eines Mail-Servers will die korrekte Benutzung seines Servers sicherstellen, indem er ihn so konfiguriert, daß ausschließlich lokale Nutzer E-Mails versenden dürfen. Um die Wirksamkeit seiner Maßnahmen zu prüfen, wertet er das Transaktionsprotokoll des Servers aus. Zu diesem Zweck teilt er Absender und Empfänger einer Nachricht in jeweils zwei Gruppen ein, interne und externe Absender ($S_I$ und $S_E$) und interne und externe Empfänger ($R_I$ und $R_E$), und leitet Kommunikationsmuster der Form $\{X_Z\} \Rightarrow \{Y_Z\}$ ab, wobei $X, Y \in (S, R), Z \in (I, E)$, jedoch mit der Einschränkung, daß $X \neq Y \; \forall \; X = S$ bzw. $X \neq Y \; \forall \; Y = S$.

Alle Muster, die einen internen Absender enthalten, sind prinzipiell unkritisch. Dagegen sind alle Muster, die einen externen Absender enthalten, kritisch, es sei denn, daß sie mindestens einen lokalen Empfänger aufweisen, d. h., es sollte immer $s(S_E) = s(S_E \cup R_I)$ und somit $c(S_E \Rightarrow R_I) = 100\%$ gelten. Das Muster $S_E \Rightarrow R_E$ ist ebenfalls problematisch. Es kann jedoch nicht verboten werden (z. B. durch $s(S_E \cup R_E) = 0$), da auch Nachrichten an externe *und* interne Empfänger geschickt werden. Um sicherzustellen, daß es in jedem Fall auch einen internen Empfänger gibt, legt der Betreiber des Servers fest, daß $s(S_E \cup R_E) = s(S_E \cup R_E \cup R_I)$ gilt. In jeder Periode muß somit lediglich überprüft werden, ob die folgenden Bedingungen erfüllt sind:

| *Muster* | *Bedingung* |
|---|---|
| $S_E \Rightarrow R_I$ | $c(S_E \Rightarrow R_I) = 100\%$ |
| $S_E \Rightarrow R_E$ | $s(S_E \cup R_E) = s(S_E \cup R_E \cup R_I)$ |

Änderungen, die zur Verletzung der genannten Bedingungen führen, werden dann dem Nutzer gemeldet. ◇

Der grundlegende Gedanke besteht darin, daß die ausgewählten Muster nicht per se interessant sind, sondern sich ihnen das Interesse des Nutzers erst zuwendet, wenn ihre Dynamik eine bestimmte Ausprägung annimmt. Natürlich sind die angegebenen Muster für den jeweiligen Anwendungskontext von größerem Interesse als andere, nicht ausgewählte Muster. Trotzdem werden dem Nutzer nicht alle ihre Änderungen gemeldet, sondern nur solche, die er durch die formulierten Bedingungen als interessant qualifiziert hat.

Der Vorteil dieser Methode liegt darin, daß ausnahmslos Änderungen gemeldet werden, die der Nutzer für interessant hält. Andererseits ist die manuelle Auswahl der Muster und die Festlegung der Grenzen für die statistischen Eigenschaften sehr zeitaufwendig und erfordert die Mitarbeit eines Anwendungsexperten. Hinzu kommt, daß ein Fehler in dieser Phase ernsthafte Auswirkungen haben kann.

### 5.3.3 Objektive Ermittlung des Interesses

Bei der subjektiven Ermittlung erweist es sich ferner als Nachteil, daß das Auswahlverfahren für die Muster bzw. die Festlegung der Grenzen der statistischen Maßzahlen in jedem Anwendungsfall erneut durchgeführt werden muß, da die Resultate dieses Prozesses meist nicht übertragbar sind. Bei der Ermittlung des Interesses nach objektiven Kriterien tritt diese Problematik

etwas in den Hintergrund. Während bei den bisher vorgestellten Ansätzen die Wahl der interessanten Musteränderungen zumindest im ersten Schritt über die Auswahl der Muster erfolgte, beziehen sich die objektiven Methoden ausschließlich auf die *Stärke* der beobachteten Änderung. Die Wichtigkeit einer Musteränderung wird nicht mehr statisch, d. h. anhand des Musters, bestimmt, sondern nur noch aufgrund der dynamischen Aspekte des Musters.

### Absolute und relative Änderungen

Bei dieser Methode gibt der Nutzer die Stärke der Änderungen an, die er für interessant hält. Dazu legt er Grenzen für den Betrag einer Musteränderung fest, deren Überschreitung er für interessant hält. Alle beobachteten Änderungen, die stärker als die vorgegebenen Grenzen sind, werden dem Nutzer gemeldet [Baron und Spiliopoulou 2001]. Im einzelnen legt er für jede beobachtete statistische Eigenschaft des Musters fest, wie hoch der absolute oder relative Betrag der Änderung zwischen zwei aufeinanderfolgenden Perioden sein muß, damit die Änderung für ihn interessant ist und gemeldet werden sollte. Diese Grenzen sollten in Übereinstimmung mit den Zielen der Analyse entwickelt werden. Außerdem sollten sie bei Bedarf, zumindest jedoch periodisch überprüft werden.

Prinzipiell sind natürlich auch andere Varianten denkbar. Sind z. B. nicht nur Muster von Interesse, die sehr starke Änderungen aufweisen, sondern auch Muster, die über längere Zeit eine gewisse Stabilität zeigen, können auch untere Grenzen für die Änderung festgelegt werden. In diesem Fall würden zusätzlich jene Änderungen angezeigt werden, deren Betrag kleiner als die vorgegebene Grenze ist. Aber auch eine Kombination von absoluter und relativer Änderung kann hilfreich sein, wenn die Verwendung nur einer Grenze noch zu viele uninteressante Änderungen liefert. Nicht zuletzt läßt sich diese Methode natürlich auch auf die Menge der interessanten Muster anwenden, die z. B. unter Verwendung des im Abschnitt 5.3.1 vorgestellten Ansatzes ermittelt wurde.

Ohne Frage ist die beschriebene Vorgehensweise sehr einfach, was ihr oft den Vorwurf der Naivität einträgt. Auf der anderen Seite kann jedoch gerade die Kombination von absoluten und relativen Grenzen zu sehr guten Resultaten führen. Darüber hinaus ließe sich eine Implementierung auf Basis dieses Prinzips leicht realisieren und wäre problemlos in existierende Lösungen zu integrieren.

### Signifikanz von Änderungen

Für den Anwender ist es vielfach von besonderem Interesse, ob bestimmte Änderungen nur zufällige Schwankungen darstellen oder ob sie tatsächlich signifikant sind, d. h. der Wert einer statistischen Eigenschaft des Musters signifikant vom Wert der Vorperiode abweicht. Für die Bewertung von Musteränderungen, d. h. für die Beantwortung der Frage, ob eine gegebene Musteränderung für interessant gehalten werden sollte, ist dieses Konzept offenbar gut geeignet: Sofern eine gegebene Änderung nicht signifikant ist, also nur eine zufällige Abweichung darstellt, ist sie auch nicht interessant.

Für die Bestimmung, ob eine Musteränderung signifikant ist, wird ein zweiseitiger Binomialtest (Test auf Anteilswert) verwendet [Baron und Spiliopoulou 2003]. Für ein gegebenes Muster $\xi$ wird geprüft, ob der Wert einer statistischen Eigenschaft $s$ zum Zeitpunkt $t_i$ für ein gegebenes Signifikanzniveau $\alpha$ signifikant vom Wert der Vorperiode abweicht:

$$H_0: \quad s_{i-1}(\xi) = s_i(\xi)$$
$$H_1: \quad s_{i-1}(\xi) \neq s_i(\xi)$$

Demnach wird jede Musteränderung als interessant betrachtet, für welche die Null-Hypothese verworfen werden muß. Dieser Test kann für alle in einer bestimmten Periode entdeckten Muster durchgeführt werden, sofern die Werte der statistischen Eigenschaften aus der Vorperiode tatsächlich vorliegen. Sind nicht alle Werte verfügbar, kann ein Mechanismus genutzt werden, wie er in [Baron u. a. 2003] vorgeschlagen wurde. Der in dieser Arbeit vorgestellte Monitor arbeitet unmittelbar auf den analysierten Daten, wobei für jedes beobachtete Muster eine Menge von SQL-Anfragen generiert wird, die die Statistiken für das Muster direkt bestimmen (vgl. Abschnitt 4.3.2). Wird ein solcher Ansatz in jeder Periode verwendet, könnten z. B. auch konventionelle Werkzeuge der Zeitreihenanalyse eingesetzt werden.

**Beispiel 5.4** Sei $s_{i-1}(\xi) = 0{,}1825$ der Support eines Musters $\xi$, das im Datensatz $D_{i-1}$ gefunden wurde, welcher im Zeitraum $t_{i-1} - t_{i-2}$ gesammelt wurde. In der Folgeperiode $t_i$ betrage die Anzahl der Transaktionen im Datensatz $D_i$, die das Muster $\xi$ unterstützen, 608, und die Gesamtzahl der Transaktionen in der Periode $t_i$ sei 2914. Es wird die Hypothese $H_0$ getestet, die besagt, daß der Support des Musters sich *nicht* geändert hat. Bei einem Signifikanzniveau von $\alpha = 0{,}01$ liegt der Nichtablehnungsbereich von $H_0$ zwischen 0,1896 und 0,2287. Da der tatsächliche Wert $s_{i-1}(\xi) = 0{,}1825$ kleiner ist als die untere Grenze des Konfidenzintervalls, wird $H_0$ verworfen und festgestellt, daß sich

der Support des Musters $\xi$ von Periode $t_{i-1}$ zu Periode $t_i$ signifikant geändert hat. ◇

Bei Tests über unbekannte Parameter wird im Rahmen der Null-Hypothese stets von der Gleichheit des vermuteten und des tatsächlichen Parameterwertes ausgegangen. Ziel des vorgestellten Ansatzes ist es, die uninteressanten Musteränderungen zu filtern, d. h. zufällige Schwankungen zu eliminieren. Es soll vermieden werden, dem Nutzer eine signifikante Änderung zu melden, obwohl es sich eigentlich um eine zufällige Schwankung handelt (fälschliche Ablehnung der Null-Hypothese, d. h. Fehler 1. Art). Bei statistischen Tests bezeichnet der Parameter $\alpha$ die Wahrscheinlichkeit, einen Fehler 1. Art zu begehen. Um dieses Risiko so gering wie möglich zu halten, sollte also ein möglichst kleiner Wert für $\alpha$ gewählt werden. Steigt $\alpha$, verkleinert sich das Konfidenzintervall, d. h. der Nichtablehnungsbereich von $H_0$, und damit die Wahrscheinlichkeit, $H_0$ zu akzeptieren. Der Nutzer könnte $\alpha$ somit als Parameter verwenden, um die Menge der gemeldeten Musteränderungen weiter zu reduzieren oder auch zu erhöhen.[1]

Dieser Test ließe sich auch auf Basis der Konfidenz verwenden. In diesem Fall würde geprüft, ob sich der Anteil der Transaktionen, die das Muster unterstützen, bezogen auf die Anzahl der Transaktionen, die die linke Seite (*Body*) des Musters unterstützen, signifikant geändert hat (vgl. Abschnitt 2.2). Natürlich ließen sich auch andere Testverfahren verwenden, um die Signifikanz von Musteränderungen zu verifizieren. So wird in [Liu u. a. 2001a] ein $\chi^2$-Test verwendet, um die Homogenität der Datensätze zweier Perioden zu prüfen (vgl. Abschnitt 3.3.1). Der wesentliche Vorteil des hier verwendeten Binomialtests ist, daß die benötigte Teststatistik direkt aus den Mining-Ergebnissen hervorgeht bzw. der Regelbasis entnommen werden kann, sofern die Ergebnisse gemäß dem temporalen Regelmodell organisiert sind (vgl. Abschnitte 4.2.1 und 4.3.2). Außerdem wird in diesem Fall ein zweiseitiger Test verwendet, mit dem signifikante Abweichungen unabhängig von ihrer Richtung ermittelt werden können.

Geht es darum, die Zufälligkeit von Musteränderungen im statistischen Sinne auszuschließen, stellen Signifikanztests das Mittel der Wahl dar. Für ein gegebenes Signifikanzniveau wird ermittelt, ob eine Abweichung tatsächlich signifikant ist oder lediglich eine zufällige Schwankung darstellt.[2] Geht es

[1] Hätte der Nutzer im Beispiel 5.4 das extreme Signifikanzniveau von $\alpha = 0{,}00001$ verwendet, wäre $H_0$ beispielsweise nicht verworfen worden.

[2] In der Statistik wird das Ergebnis eines solchen Tests deutlich vorsichtiger interpre-

zusätzlich jedoch auch um die Frage, welche zeitliche Dimension die Änderung aufweist, kann ein einfacher Signifikanztest nur Bestandteil der Analyse sein. So ist es vorstellbar, daß eine langfristige Änderung sich nur langsam, d. h. über einen Zeitraum von mehreren Perioden, manifestiert. Jede einzelne Änderung für sich genommen muß dabei nicht notwendigerweise immer auch signifikant sein. Wie sich bei der experimentellen Überprüfung der im folgenden Abschnitt vorgestellten Heuristiken zur Ermittlung interessanter Musteränderungen gezeigt hat, können Signifikanztests zudem auch nicht alle interessanten Änderungen ermitteln (vgl. Abschnitt 7.2).

Wie auch bei der Anwendung anderer statistischer Testverfahren stellt sich die Frage, ob dem Test zugrunde liegende Annahmen verletzt sind. Sollen statistische Tests zur Prüfung der Signifikanz einer Änderung verwendet werden, so muß z. B. die *iid*-Voraussetzung erfüllt sein, d. h., die beobachteten Zufallsvariablen müssen unabhängig und identisch verteilt sein (engl. *independently and identically distributed*, abgekürzt *iid*). Diese fundamentale Annahme ist jedoch sicherlich nicht immer gerechtfertigt. Ein zusätzliches Problem ergibt sich aus der potentiell unendlichen Anzahl betrachteter Perioden. Sieht man den Test in jeder Periode als Zufallsexperiment, ergeben sich für die beiden möglichen Ergebnisse $X =$ „$H_0$ nicht verwerfen in Periode $t_i$“ und $\bar{X} =$ „$H_0$ verwerfen in Periode $t_i$“ die folgenden Wahrscheinlichkeiten:

$$\begin{aligned} P(X) &= 1-\alpha \qquad \text{bzw.} \\ P(\bar{X}) &= 1-P(X) \end{aligned}$$

Sind die Tests unabhängig voneinander, bestimmt sich die Wahrscheinlichkeit für das Ereignis $Y =$ „$H_0$ nicht verworfen in $n$ Perioden“ aus

$$P(Y) = (1-\alpha)^n,$$

und für große Werte von $n$ ergibt sich

$$\lim_{n\to\infty} P(Y) = 0.$$

Die gleiche Betrachtung für die Ablehnung von $H_0$ in Periode $n$ zeigt, daß

$$\begin{aligned} P(\bar{Y}) &= 1-P(Y) \\ &= 1-(1-\alpha)^n \end{aligned}$$

tiert. Hier wird im allgemeinen nur festgestellt, daß es ausgehend von der vorliegenden Zufallsstichprobe und den verwendeten Parametern statistisch gesehen keinen Anlaß gab, die Null-Hypothese abzulehnen bzw. beizubehalten.

und damit für die unendliche Betrachtung

$$\lim_{n\to\infty} P(\bar{Y}) = 1$$

gilt. Das bedeutet, daß die Null-Hypothese *irgendwann* ungerechtfertigt abgelehnt wird. Da aufgrund der unbekannten Periodenzahl auch die beim Testen multipler Hypothesen übliche Anpassung des Parameters $\alpha$ nicht vorgenommen werden kann, bleibt die Anwendung von Signifikanztests mit einem gewissen Risiko verbunden.

### 5.3.4 Heuristiken zur Ermittlung des Interesses

In diesem Abschnitt werden drei Heuristiken beschrieben, die im wesentlichen auf die Arbeit, die in [Baron und Spiliopoulou 2003] vorgestellt wurde, zurückgehen. Das Ziel dieser Heuristiken ist ebenfalls, interessante Musteränderungen zu erkennen. Und obwohl sie gleichermaßen auf objektiven Kriterien beruhen, beziehen sie im Gegensatz zu den bisher besprochenen Verfahren zusätzlich unterschiedliche Aspekte der *Stabilität* von Mustern ein. Allen drei Verfahren ist dabei gemeinsam, daß sie nach Abweichungen von der *erwarteten* Entwicklung der statistischen Eigenschaften der Muster suchen – sowohl auf der Ebene der gesamten Regelbasis als auch einer einzelnen Regel.

#### Stabilitätsbasierte Gruppierung

Alle Muster, die in einer bestimmten Periode erscheinen, spiegeln die Zusammenhänge innerhalb des Datensatzes wider, die charakteristisch für diese Periode sind. Demnach sollten Muster, die in allen Perioden präsent sind, die invarianten Eigenschaften der Daten repräsentieren, zumindest jedoch Teile davon. Im Abschnitt 5.2.2 gruppieren wir deswegen Muster anhand der *Stabilität*, die sie im Zeitverlauf zeigen. Stabilität wird dabei als der Anteil der Perioden bestimmt, in denen ein gegebenes Muster sichtbar war, d. h. die in der Mining-Anfrage angegebenen Schwellen für z. B. Support und Konfidenz überschritten hat, bezogen auf die Gesamtanzahl der Perioden. Muster, die eine vergleichbare Stabilität zeigen, werden zu Gruppen zusammengefaßt ($P, H+, H, M$ und $L$).

In jeder Periode $t_i$ werden die Muster – entweder alle oder eine ausgewählte zu beobachtende Menge – aufgrund ihrer in der Vergangenheit gezeigten Frequenz einer der festgelegten Gruppen zugeordnet. Anschließend

wird überprüft, ob eines der Muster seit der letzten Zuordnung die Gruppe gewechselt hat. Gehört ein Muster $\xi$ in zwei aufeinanderfolgenden Perioden $t_{i-1}$ und $t_i$ zwei unterschiedlichen Gruppen an, wird dem Nutzer für dieses Muster eine interessante Änderung signalisiert. Parallel dazu werden die Änderungen der Gruppen verfolgt. Im einzelnen wird für jede Gruppe angegeben, wieviel Regeln gegenwärtig in ihr enthalten sind, wieviel Regeln seit der letzten Mining-Session hinzugekommen sind und wie viele die Gruppe verlassen haben.

Für den Nutzer sind diese Informationen wesentlich. Auf der einen Seite werden Änderungen auf der Ebene eines einzelnen Musters gemeldet. Diese Änderung ist jedoch langfristiger Natur. Sie betrifft weniger die momentanen Ausprägungen der statistischen Eigenschaften, sondern die Stabilität der Ausprägungen im Zeitverlauf. Wenn ein Muster die Gruppe wechselt, kann das natürlich auch mit starken Änderungen seiner Statistiken verbunden sein. Insbesondere in späten Perioden langwieriger Analysen kann jedoch auch der dann wahrscheinlichere Fall eintreten, daß das Muster in der Periode, in der der Gruppenwechsel stattgefunden hat, überhaupt keine starken Veränderungen zeigt. Der Gruppenwechsel ist vielmehr nur Ausdruck der Tatsache, daß es nicht mehr so häufig in den Ergebnissen vertreten ist.

**Beispiel 5.5** Es wird die Präsenz eines Musters $\xi$ über einen Zeitraum von insgesamt 200 Perioden beobachtet. Der minimale Support, der in der Mining-Anfrage angegeben wurde, betrage $\tau_s = 20\%$. Während das Muster zunächst in allen Perioden vertreten ist, kann es ab Periode $t_{60}$ in jeder sechsten Periode nicht mehr gefunden werden. Von Periode $t_{100}$ bis Periode $t_{140}$ ist es überhaupt nicht sichtbar, um dann wieder in jeder Periode zu erscheinen. Abbildung 5.1 zeigt die Entwicklung der Frequenz des Musters $f(\xi)$ über alle Perioden. Nachdem das Muster in Periode $t_{60}$ die Gruppe der permanenten Muster verlassen hat, wechselt es in Periode $t_{103}$ in die Gruppe $H$ und in Periode $t_{124}$ in die Gruppe $M$. In Periode $t_{191}$ wechselt das Muster erneut in die Gruppe $H$. Interessant sind insbesondere die letzten beiden Gruppenwechsel. Nach Periode $t_{100}$ gibt es bezüglich der statistischen Eigenschaften eigentlich nur in Periode $t_{141}$ ein interessantes Ereignis – der Support des Musters lag erstmals wieder über der Grenze von 20%, nachdem er seit Periode $t_{100}$ darunter lag. In den verbleibenden Perioden liegt der Support dann wieder dauerhaft über 20%. Trotzdem wird in den Perioden $t_{124}$ und $t_{191}$ eine interessante Musteränderung signalisiert. Aber diese Änderung betrifft eben nicht die gegenwärtigen Werte der statistischen Eigenschaften, sondern

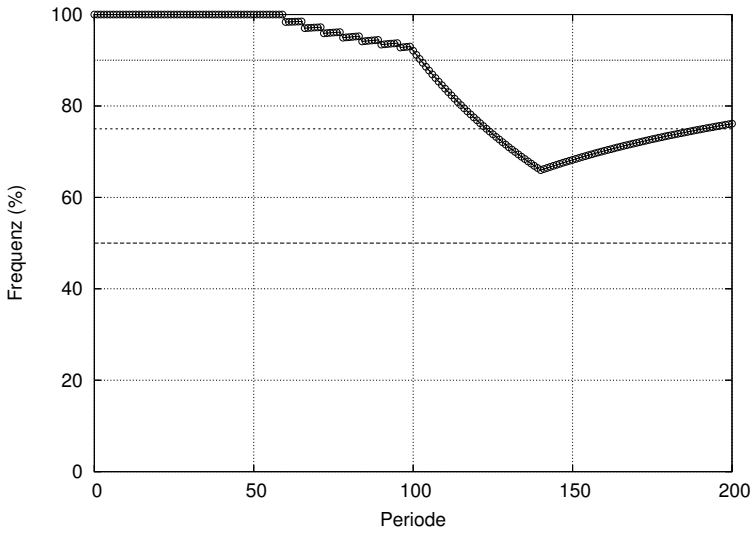


Abbildung 5.1: Frequenz eines Musters $f(\xi)$

die gegenwärtige Stablilität, mit der die in der Mining-Anfrage spezifizierten Schwellen bis zum jeweils betrachteten Zeitpunkt überschritten wurden. ◇

Zusätzlich wird aber auch die Entwicklung der verschiedenen Gruppen über die Zeit gezeigt, so daß der Nutzer zu jedem Zeitpunkt einen Eindruck erhält, zu welchen Teilen die Menge der gefundenen Muster stabil oder instabil ist, d. h., welchen Support die Gruppen bezogen auf die gesamte Regelbasis haben. In diesem Fall werden nicht spezifische Regeln betrachtet, sondern die Auswirkungen auf die Regelbasis, die mit Änderungen der Regeln einhergehen.

Natürlich kann diese Heuristik nur nach einer ausreichenden Anzahl von Trainingsperioden brauchbare Resultate liefern, da sie in den ersten Perioden sehr empfindlich auf Änderungen der Regelbasis reagiert. Im Gegensatz zum Signifikanztest, der im allgemeinen nur für sichtbare Muster durchgeführt werden kann, wird die Zuordnung der Muster zu den jeweiligen Gruppen ab der ersten Periode ihres *Erscheinens* vorgenommen. Nach einer Lernphase, deren Länge vom Anwendungskontext abhängt, werden dem Anwender dann eventuelle Gruppenänderungen signalisiert. Zusätzlich zu der im Beispiel 5.5 beschriebenen Eigenschaft ist anzunehmen, daß die Heuristik interessante Änderungen identifizieren kann, die nicht durch Signifikanztests ermittelt

werden könnten. Das liegt daran, daß Änderungen der Regelbasis, d. h. das Verschwinden bzw. Erscheinen eines Musters, häufig mit starken Änderungen der statistischen Eigenschaften des Musters verbunden sind, die durch Signifikanztests nicht ohne weiteres erkannt werden können, da das Muster in den jeweiligen Perioden nicht vorhanden ist. Allerdings sollte sich dieser Effekt mit zunehmender Länge des Analysezeitraums weniger stark bemerkbar machen (vgl. Beispiel 5.5).

### Korridorbasierte Heuristik

Diese Heuristik definiert einen *Korridor* um die Zeitreihe einer statistischen Eigenschaft des Musters. Dieser Korridor entspricht einem Intervall von Werten und wird in jeder Periode angepaßt – zu jedem Zeitpunkt $t_i$ werden somit alle bisherigen Werte der betrachteten Eigenschaft einbezogen. Für ein Muster $\xi$ und eine statistische Eigenschaft $s$ berechnen wir den Mittelwert $\mu_i$ und die Standardabweichung $\sigma_i$ der Werte $\{s_j(\xi)|j = 0, 1, \ldots, i\}$. Der Korridor zum Zeitpunkt $t_i$ ist definiert als das Intervall $I(t_i) := [\mu_i - \sigma_i/2, \mu_i + \sigma_i/2]$, mit einer Breite von $\sigma_i$ und dem Mittelpunkt $\mu_i$. Eine interessante Änderung wird immer dann signalisiert, wenn für ein Muster der Wert der Zeitreihe $s_i(\xi)$ außerhalb des Korridors $I(t_i)$ liegt. Im Ergebnis wird diese Heuristik also genau dann eine interessante Änderung melden, wenn die Abweichung einer statistischen Eigenschaft vom Wert der Vorperiode größer ausfällt, als aufgrund der in vergangenen Perioden gesammelten Erfahrungen anzunehmen ist.

Natürlich könnte man zur Bestimmung des Korridors auch andere statistische Maßzahlen, z. B. Median und Quantile, verwenden und/oder die Breite des Korridors modifizieren. Der prinzipielle Vorteil der beschriebenen Methode ist jedoch, daß sie genau jene Musteränderungen meldet, die man *intuitiv* als interessant bezeichnen würde – unerwartete Abweichungen vom sonst üblichen Niveau. Darüber hinaus ist der Korridor einerseits unempfindlich gegenüber Schwankungen um die Schwellen, die für die Entdeckung der Muster verwendet wurden, andererseits aber reagiert er auch auf Änderungen, die zwar von früheren Werten abweichen, jedoch immer noch im Intervall liegen.

Selbstverständlich lassen sich der Mittelwert und die Standardabweichung nur nach einer hinreichend großen Anzahl von Perioden sinnvoll bestimmen. Deswegen sollte die Identifizierung interessanter Musteränderungen erst nach einer ausreichend langen Trainingsperiode bzw. nur im Rahmen einer retro-

spektiven Studie erfolgen. Ist die Anzahl der Perioden dagegen sehr groß, sollte unter Umständen die Verwendung eines gleitenden Zeitfensters in Betracht gezogen werden, da historische Werte die Zeitreihen sonst eventuell zu stark beeinflussen und die Ergebnisse verfälschen. Auch wenn interessante Änderungen erst nach der Trainingsphase signalisiert werden, beginnt die Überwachung eines Musters – wie im Fall der stabilitätsbasierten Heuristik – mit seinem ersten Auftreten, d. h. ab der ersten Periode, in der es gefunden wurde. Die Berechnung des Korridors ist dabei natürlich nur möglich, wenn *alle* Folgeperioden einbezogen werden, weswegen die Mining-Software unbeschränkt arbeiten oder der im Abschnitt 4.3.2 vorgestellte Ansatz verfolgt werden sollte.[3] Auch in diesem Fall ist infolgedessen damit zu rechnen, daß interessante Änderungen, die nicht durch Signifikanztests ermittelt werden können, entdeckt werden. Haben solche Änderungen Auswirkungen auf die Sichtbarkeit des Musters, so sind sie im allgemeinen von besonderem Interesse für den Nutzer, kann er doch auf diese Weise den Grund für das Verschwinden eines Musters nachvollziehen.

Eine vereinfachte Variante dieses Verfahrens, die natürlich auch zusätzlich einbezogen werden kann, beschränkt sich darauf, lediglich die Breite des Korridors, d. h. die Standardabweichung der Zeitreihe, im Zeitverlauf zu überwachen, und vernachlässigt die Frage, ob der konkrete Wert $s_i(\xi)$ innerhalb des Korridors liegt oder nicht. Für längerfristige Analysen könnte zusätzlich geprüft werden, ob signifikante Änderungen des Mittelwertes und/oder der Varianz festzustellen sind.

### Intervallbasierte Heuristik

Bei dieser Heuristik wird der Wertebereich einer statistischen Eigenschaft $s$ eines Musters $[\tau_s, M]$ in $k$ Intervalle mit gleicher Breite eingeteilt, wobei $M$ der maximal zulässige Wert für die jeweils beobachtete statistische Eigenschaft ist und $\tau_s$ entweder der in der Mining-Anfrage verwendeten Schwelle entspricht oder der minimal zulässige Wert für die jeweils beobachtete sta-

[3] Unter der unbeschränkten Arbeitsweise eines Miners verstehen wir die fehlende Limitierung in der Mining-Anfrage, die in [Baron und Spiliopoulou 2001] dargestellt wurde. Verfolgt man einen solchen Ansatz, wird eigentlich keine Mining-Anfrage gestellt, sondern es werden *alle vorhandenen* Zusammenhänge in den Daten entdeckt. Dies ließe sich beispielsweise erreichen, indem in der Mining-Anfrage die Grenzen für Support und Konfidenz auf $\tau_s = 0\%$ und $\tau_c = 0\%$ festgelegt werden. Allerdings ist bei dieser Vorgehensweise mit fatalen Konsequenzen für die Bearbeitungszeit der Anfrage zu rechnen.

tistische Eigenschaft ist. Eine interessante Änderung wird prinzipiell immer dann signalisiert, wenn für ein Muster $\xi$ der Wert der Zeitreihe zum Zeitpunkt $t_i$ in einem anderen Intervall liegt als im Zeitpunkt $t_{i-1}$. Zusätzlich wird bei einem Intervallwechsel geprüft, wie stark die absolute Änderung war, die zu dem Wechsel geführt hat. Neben der Anzahl von Intervallen $k$ gibt der Nutzer dazu einen optionalen Parameter $\epsilon$ an, der die untere Grenze für die Stärke der Änderung spezifiziert. Unter Verwendung von $\epsilon$ wird eine interessante Änderung nur dann signalisiert, wenn die absolute Änderung, die zum Intervallwechsel geführt hat, mindestens so groß wie $\epsilon$ ist. Auf diese Weise wird die Heuristik unempfindlich für minimale Änderungen an den Intervallgrenzen. Andererseits wird auch dann eine interessante Änderung gemeldet, wenn sie größer ist als $(M - \tau_s)/k - \epsilon$ und kein Intervallwechsel vorliegt. Dies ist insbesondere dann wichtig, wenn $k$ klein ist, d. h. $(M-\tau_s)/k$ ein vergleichsweise großes Intervall darstellt, und eine starke Änderung nicht auch zu einem Intervallwechsel führt.

**Beispiel 5.6** Für das Muster $\xi$ wird die aus Tabelle 5.1 ersichtliche Zeitreihe seiner Konfidenz untersucht. In der Mining-Anfrage wurde die Schwelle für die Konfidenz auf $\tau_c = 0{,}75$, die Anzahl der Intervalle auf $k = 4$ und das absolute Minimum für eine Änderung auf $\epsilon = 0{,}05$ festgelegt. Für die Konfidenz ergibt sich damit ein Bereich von [0,75, 1], der in die Intervalle $I_1$ = [0,75, 0,8125), $I_2$ = [0,8125, 0,875), $I_3$ = [0,875, 0,9375) und $I_4$ = [0,9375, 1] aufgeteilt wird. Für jeden Zeitpunkt ist angegeben, in welches Intervall der Wert

| | | Intervall | | | | | |
|---|---|---|---|---|---|---|---|
| Periode | Konfidenz | $I_1$ | $I_2$ | $I_3$ | $I_4$ | Wechsel | Änderung |
| $t_0$ | 0,75 | √ | | | | | |
| $t_1$ | 0,82 | | √ | | | √ | √ |
| $t_2$ | 0,87 | | √ | | | – | √ |
| $t_3$ | 0,88 | | | √ | | √ | – |

Tabelle 5.1: Zeitreihe der Konfidenz eines Musters $\xi$

der Konfidenz fällt. Die vorletzte Spalte gibt an, ob ein Intervallwechsel stattgefunden hat, die letzte Spalte, ob eine interessante Änderung signalisiert werden sollte. In Periode $t_1$ befindet sich $c(\xi)$ in einem anderen Intervall, *und*

die absolute Änderung war größer als $\epsilon$. In Periode $t_2$ findet zwar kein Intervallwechsel statt, der Wert der absoluten Änderung ist jedoch nicht kleiner als $\epsilon$, weswegen eine interessante Änderung signalisiert wird. In der letzten Periode führt eine relativ kleine Änderung zu einem Intervallwechsel. Weil die absolute Änderung kleiner als $\epsilon$ ist, wird jedoch keine interessante Änderung signalisiert. ◇

Es ist ersichtlich, daß diese Methode ebenso wie die Signifikanztests bereits ab der zweiten Periode angewandt werden kann. Die Anzahl der Intervalle $k$ und die Größe der absoluten Änderung $\epsilon$ sind dabei Parameter, die an die spezifischen Bedürfnisse der jeweiligen Anwendung angepaßt werden können. Auf Basis der Intervalle ist zusätzlich die Bildung von Gruppen denkbar, deren Analyse bzw. Überwachung Aufschlüsse hinsichtlich der Stabilität der Ausprägungen der erfaßten statistischen Maßzahlen liefern kann.

## 5.4 Zeitliche Dimension von Musteränderungen

Sind die Änderungen, denen die Muster in der Regelbasis im Zeitverlauf ausgesetzt sind, erst einmal bekannt, stellt sich die Frage, welche zeitliche Dimension sie aufweisen. Für den Anwender ist es natürlich von Interesse, ob eine Änderung, die er für interessant hält, dauerhaft ist und damit grundlegende Änderungen in der untersuchten Population repräsentiert oder ob sie eine kurzfristige Abweichung darstellt. Möglicherweise ließen sich zur Beantwortung der Frage, ob eine bestimmte Änderung Teil eines langfristigen Trends ist, konventionelle Werkzeuge der Zeitreihenanalyse verwenden, wobei jedoch genau überprüft werden müßte, ob die hierbei üblichen Annahmen in bezug auf die beobachtete Population zutreffend sind (vgl. Abschnitt 3.5). Für die Überwachung von Mustern stellt sich darüber hinaus jedoch auch das Problem ihrer *Sichtbarkeit*. Erfüllt ein in einer früheren Periode gefundenes Muster nicht mehr die in der Mining-Anfrage vorgegebenen Bedingungen, ist es nicht länger sichtbar, und die Zeitreihen der beobachteten statistischen Eigenschaften sind unterbrochen. In diesem Fall sind Standardverfahren der Trendanalyse nicht ohne weiteres anwendbar. Zusätzlich ergibt sich die Problematik, daß derartige Verfahren oft eine Glättung der Zeitreihe vornehmen, um Trends abzuleiten. Starke Änderungen, die im hier betrachteten Kontext von besonderem Interesse wären, gehen dadurch verloren. Dem er-

sten Problem kann – wie bereits angesprochen – durch die Verwendung des im Abschnitt 4.3.2 beschriebenen SQL-basierten Monitors entgegengetreten werden. Das zweite Problem schließt die Nutzung solcher Verfahren nicht prinzipiell aus. Vielmehr könnte eine solche Analyse parallel durchgeführt werden, um zusätzliche Informationen bereitzustellen.

Das in diesem Abschnitt vorgestellte Verfahren bietet die Möglichkeit, die temporale Dimension einer Änderung mit begrenztem Aufwand zu bestimmen. Ein wesentlicher Vorteil liegt darin, daß es bereits mit Beginn der Überwachung der Muster eingesetzt werden kann und nicht wie bei Trendanalysen auf eine ausreichend große Zahl an Perioden angewiesen ist. Abgesehen von retrospektiven Analysen, bei denen die Aktualität der Ergebnisse nicht das wesentliche Ziel ist, sind zeitnahe Informationen für den Anwender immer dann besonders wichtig, wenn er sich einen ausreichenden Handlungsspielraum bewahren will. Während eine Zeitreihenanalyse einen Trend aber erst bestimmen kann, wenn er sich deutlich in den Daten manifestiert hat, kann das vorgeschlagene Verfahren bereits in einem relativ frühen Stadium dazu erste Hinweise liefern.

### 5.4.1 Kurz- und langfristige Musteränderungen

Für jede interessante Musteränderung, die nach einem der in den letzten Abschnitten beschriebenen Verfahren identifiziert wurde, wird geprüft, ob sie eine kurzfristige Änderung darstellt, die lediglich temporär zu beobachten war, oder längerfristigen Charakter hat, d. h. der Wert der jeweils betrachteten statistischen Eigenschaft des Musters nicht unmittelbar zu seinem ursprünglichen Niveau zurückkehrt. Zu diesem Zweck wird vom Nutzer ein Parameter $m$ angegeben, der bestimmt, wie viele Perioden die beobachtete Änderung währen muß, um als langfristige Änderung angesehen zu werden. Andererseits werden Änderungen, die vor Ablauf von $m$ Perioden wieder aufgehoben sind, als kurzfristige Änderungen bezeichnet.

Im einzelnen folgt daraus für die interessanten Änderungen, die sich aufgrund eines Signifikanztests in Periode $t_i$ ergeben haben, daß im Zeitraum $t_{i+1}$ bis $t_{i+m}$ überprüft wird, ob eine signifikante Änderung für die betrachtete statistische Eigenschaft beobachtet werden kann, die die entgegengesetzte Richtung wie die ursprüngliche Änderung in $t_i$ aufweist und diese wieder aufhebt. Kann eine derartige Änderung entdeckt werden, wird die Änderung in $t_i$ als kurzfristig gekennzeichnet, anderenfalls als langfristig. In gleicher Weise wird bei der korridor- und intervallbasierten Heuristik vorgegangen.

Wird eine Verletzung des Korridors bzw. ein Intervallwechsel signalisiert, findet in den nächsten $m$ Perioden die Prüfung statt, ob der Wert in den Korridor bzw. das ursprüngliche Intervall zurückkehrt. Dabei ist der Fall der korridorbasierten Heuristik von besonderem Interesse. Selbst wenn der Wert der statistischen Eigenschaft nicht zu seinem ursprünglichen Niveau zurückkehrt, ist damit zu rechnen, daß er früher oder später wieder innerhalb des Korridors liegt, da die aktuellen Werte der Zeitreihe in die Ermittlung des Korridors einbezogen werden. Aus diesem Grund deuten langfristige Änderungen, die mit Hilfe der korridorbasierten Heuristik identifiziert wurden, auf wesentliche Änderungen in der untersuchten Population hin und sollten genauer untersucht werden.

**Beispiel 5.7** Es wird die in Abbildung 5.2 dargestellte Zeitreihe der Konfidenz eines Musters $\xi$ betrachtet. Die waagerechte Linie bei 0,8 bezeichnet die Schwelle der Konfidenz ($\tau_c = 80\%$), die im Rahmen der Musterentdeckung verwendet wurde. Die senkrechte Linie zum Zeitpunkt $t_{40}$ bezeichnet das Ende der Trainingsphase. Die Werte der Zeitreihe liegen in den Perioden $t_{58}$, $t_{64}$

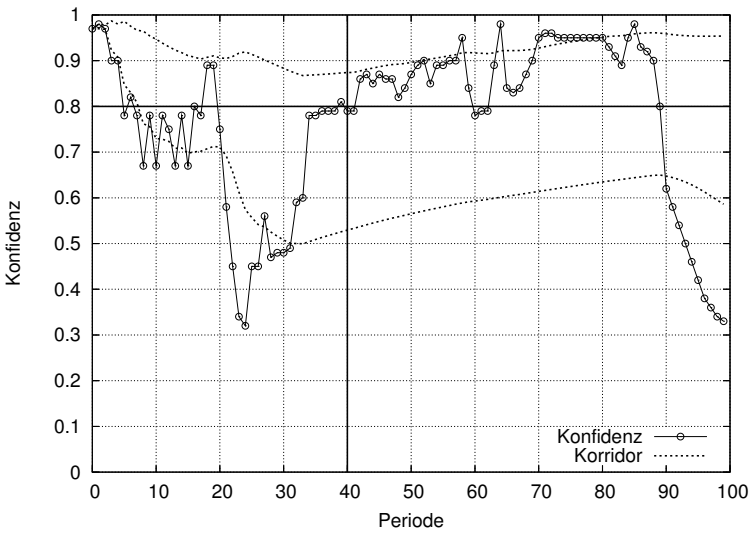


Abbildung 5.2: Konfidenz eines Musters $c_i(\xi)$

und $t_{85}$ für jeweils eine Periode außerhalb des Korridors.[4] Zu diesen Zeitpunk-

[4]Aus Gründen der besseren Darstellbarkeit wurde in diesem Beispiel ein Korridor der

ten werden dem Nutzer kurzfristige interessante Änderungen signalisiert. In Periode $t_{70}$ liegt der Wert ebenfalls außerhalb des Korridors, was in diesem Fall auch für die Werte der Folgeperiode zutrifft. Ob dem Nutzer eine langfristige interessante Änderung gemeldet wird, hängt vom Wert ab, der für $m$ festgelegt wurde. Es wird allerdings deutlich, daß die Zeitreihe in Periode $t_{80}$ wieder in den Korridor zurückkehrt, obwohl sich die Konfidenz zwischen den Perioden $t_{73}$ und $t_{80}$ nicht ändert. Im Gegensatz dazu ist die Änderung der Konfidenz, die in Periode $t_{90}$ zur erneuten Verletzung des Korridors führt, so stark, daß der Korridor sich nicht schnell genug anpassen kann und die Werte der Zeitreihe dauerhaft außerhalb des Korridors liegen. ◇

Auch wenn die Anwendung dieser Methode auf die Ergebnisse der stabilitätsbasierten Gruppierung von Mustern prinzipiell möglich ist, sind insbesondere in späten Perioden langfristiger Analysen keine sehr aussagekräftigen Resultate zu erwarten. Bei einem weiten Zeithorizont wird die stabilitätsbasierte Heuristik ohnehin verstärkt Änderungen langfristiger Natur signalisieren.

Für den Anwender ist die beschriebene Methode immer dann von Vorteil, wenn die Informationen möglichst zeitnah vorliegen sollen. Die Frage, welche zeitliche Dimension eine interessante Änderung aufweist, kann bereits nach $m + 1$ Perioden approximativ beantwortet werden. Zur endgültigen Klärung der Frage, ob eine langfristige Änderung tatsächlich einen Trend widerspiegelt, liefert jedoch sicher nur eine retrospektive Untersuchung der Historie der statistischen Eigenschaften, z. B. mit Hilfe einer konventionellen Zeitreihenanalyse, ausreichenden Aufschluß.

### 5.4.2 Bewertung temporaler Änderungen

Die Bewertung der zeitlichen Dimension von Musteränderungen hängt im wesentlichen von den Zielen der Analyse ab. Im allgemeinen werden kurzfristige Abweichungen beim Anwender sicherlich auf ein geringeres Interesse stoßen als langfristige Änderungen, die Hinweise auf grundlegende Änderungen in der betrachteten Population liefern können. Eine Klärung dieser Frage ohne Bezug zum Anwendungskontext liefert sicher nur eine unbefriedigende Antwort. Allerdings ist die Klärung der temporalen Dimension auch notwendig, um die Zahl der als interessant erachteten Musteränderungen weiter zu begrenzen. Für kurzfristige Abweichungen werden dem Anwender gewöhnlich

Breite $2\sigma$ verwendet.

zwei Änderungen angezeigt – das Verlassen des ursprünglichen Niveaus in Periode $t_i$ und die Rückkehr zum alten Niveau in Periode $t_{i+1}$. Ungeachtet des Interesses, das der Anwender kurzfristigen Änderungen entgegenbringt, sollte nur die Änderung in $t_i$ signalisiert werden. Stellt sich heraus, daß die Änderung in Periode $t_{i+1}$ die Rückkehr zum ursprünglichen Niveau darstellt, sollte die Änderung in $t_i$ als kurzfristig gekennzeichnet und die Änderung in $t_{i+1}$ verworfen werden.

Auf der anderen Seite stellt sich die Frage, wie die Bewertung von Musteränderungen Einfluß auf die Auswahl der interessanten Muster nehmen kann. Speziell für retrospektive Studien ließe sich die Auswahl interessanter Regeln auch auf Basis ihrer temporalen Eigenschaften vornehmen. So könnten unter diesem Blickwinkel für den Anwender all jene Muster interessant sein, die in sämtlichen Perioden gefunden werden können oder deren statistische Eigenschaften nur geringe Schwankungen im Zeitverlauf aufweisen. Während die Bewertung der dynamischen Eigenschaften von Mustern aufgrund statischer Aspekte die im Abschnitt 5.3.1 angesprochenen Nachteile mit sich bringt, könnte die umgekehrte Vorgehensweise durchaus von Vorteil sein. Muster, die in allen Perioden vertreten sind, reflektieren grundlegende Zusammenhänge des betrachteten Datensatzes und sind für den Anwender im Normalfall von größerer Wichtigkeit als Muster, die nur sporadisch auftreten. Gleiches gilt sicherlich für Muster, die periodisch auftreten. Problematisch ist diese Vorgehensweise lediglich im Hinblick auf die Tatsache, daß eine solche Betrachtung erst nach einer hinreichend großen Anzahl von Perioden bzw. für retrospektive Analysen aussagekräftige Ergebnisse liefern kann.

Auch wenn die Frage, welche Ausprägung der zeitlichen Dimension von Musteränderungen für den Anwender von größerem Interesse ist, nicht losgelöst vom Anwendungskontext beantwortet werden kann, sollte sie nicht außer acht gelassen werden. Dem Nutzer liefert diese Betrachtung in jedem Fall zusätzliche Informationen und dürfte das Verständnis für die aus den Daten abgeleiteten Modelle verbessern. Darüber hinaus bietet sich die Möglichkeit, eine Zuordnung von Musteränderungen und nicht im Datensatz erfaßten Ereignissen vornehmen zu können: In den seltensten Fällen sind alle Faktoren, die einen Datensatz beeinflussen können, selbst darin enthalten, und zusätzliche Informationen über die temporale Dimension der Änderungen können helfen, derartige Verbindungen herzustellen.

## 5.5 Ursachen von Musteränderungen

Hat man die Menge der interessanten Änderungen bestimmt, liegt es nahe festzustellen, warum sich diese Regeln geändert haben. Externe Faktoren sind naturgemäß schwer zu bestimmen, auch wenn sie sich zeitlich gut einordnen lassen und meist deutlich an ihren Wirkungen zu erkennen sind. Aus diesem Grund ist es sinnvoll, sich auf die Ursachen zu konzentrieren, die in den Daten enthalten sind. Dazu muß ermittelt werden, welche – möglicherweise ebenfalls interessanten – Änderungen mit den betrachteten Änderungen einhergingen und ob sie diese eventuell ausgelöst haben. Dabei kommen prinzipiell inhaltliche Übereinstimmungen zwischen Mustern und Ähnlichkeiten in ihrer zeitlichen Entwicklung in Betracht. Muster, die eine ähnliche Entwicklung über die Zeit zeigen, können mit Hilfe des Ansatzes von Agrawal u. a. [1995] ermittelt werden (vgl. Abschnitt 3.5). Während mit dieser Technik ausschließlich Abhängigkeiten bestimmt werden können, die sich in ähnlichen Zeitreihen widerspiegeln, gibt es eine Reihe statistischer Verfahren, um andere Arten der Korrelation zwischen Zufallsgrößen zu erkennen.

Wir konzentrieren uns auf Ursachen, die unmittelbar aus dem untersuchten Datensatz hervorgehen, und betrachten Muster, die einander *ähnlich* sind, d. h. inhaltliche Übereinstimmungen aufweisen. Sind sich zwei Muster ähnlich, so sollte eine Änderung des Teiles, der beiden gemeinsam ist, Auswirkungen auf beide Muster haben.

**Beispiel 5.8** Es seien zwei Assoziationsregeln $AB \Rightarrow C$ und $AB \Rightarrow D$ betrachtet, die einen Zusammenhang zwischen den Produkten $A$, $B$ und $C$ bzw. $D$ beschreiben. Beide Regeln stimmen hinsichtlich ihrer Bedingung überein. Wird nun die Kombination der Produkte $A$ und $B$ sehr viel öfter verkauft, sollte dies – sofern nicht die Änderung bei einem der Muster durch eine andere Änderung aufgehoben wird – die Konfidenz beider Muster beeinflussen. Angenommen der Support des Itemsets $AB$ steigt um 10% seines ursprünglichen Wertes. Während der Support für die rechte Seite der ersten Regel ebenfalls um 10% ansteigt, bleibt der Support der rechten Seite der zweiten Regel konstant. In einem solchen Fall würde sich die Konfidenz für das zweite Muster vermindern, während sie für das erste Muster unverändert bliebe. Obwohl beide Muster die gleiche linke Seite haben, wirkt sich die Support-Änderung unterschiedlich aus. ◇

Das Beispiel deutet an, wie schwierig unter Umständen die Suche nach den Ursachen für eine Änderung sein kann. Um die Ursachen der Musteränderun-

gen zu bestimmen, werden deshalb die Itemsets, aus denen sich die Muster zusammensetzen, betrachtet. Jede der im Abschnitt 5.3.4 beschriebenen Heuristiken liefert eine Menge von interessanten Änderungen zurück. Alle Muster, die interessante Änderungen zeigen, werden in ihre *Komponenten*, d. h. die Menge aller möglichen Kombinationen der Elemente der Regel, zerlegt, welche unter Verwendung des im Abschnitt 5.3.3 vorgestellten statistischen Tests auf Signifikanz untersucht werden. Zu diesem Zweck verwenden wir den in Abbildung 5.3 dargestellen Algorithmus.

**Algorithmus** findCauses($\Xi$)**:**

```
 1  causes := ∅;
 2  for each ξ ∈ Ξ; do
 3      Ω := computeLongestComponents(ξ);
 4      for each ω ∈ Ω; do
 5          if ω ∉ causes and length(ω) > 0; then
 6              if isInteresting(ω) = true; then
 7                  causes := causes ∪ ω;
 8                  causes := causes ∪ findCauses(ω);
 9              endif;
10          endif;
11      done;
12  done;
13  return causes;
```

Abbildung 5.3: Ermittlung der Ursachen interessanter Änderungen

Das Argument bezeichnet die Menge der Muster bzw. Itemsets, die interessante Änderungen aufweisen. In Zeile 1 initialisiert der Algorithmus die Ergebnismenge. Dann wird für jedes Itemset, das interessante Änderungen aufweist, die Menge der längsten Komponenten bestimmt (Zeile 3). Für jede dieser Komponenten wird nun überprüft, ob diese Komponente bereits überprüft wurde und ob die Länge der Komponente größer als Null ist (Zeile 5). Wenn dem so ist, wird mit Hilfe des Binomialtests überprüft, ob die Komponente eine signifikante Änderung aufweist (Zeile 6). Falls es sich so verhält, wird die Komponente der Ergebnismenge hinzugefügt und in einem rekursiven Aufruf erneut an den Algorithmus übergeben, wobei der Rückgabewert der Ergebnismenge hinzugefügt wird (Zeilen 7 und 8). In der letzten Zeile wird die Ergebnismenge an die aufrufende Funktion zurückgegeben.

Es ist offensichtlich, daß die Komplexität der Überprüfung mit der Regellänge ansteigt, da die Menge der Komponenten als die Menge aller möglichen Kombinationen der Elemente der Regel definiert ist. Sei $n$ die Anzahl der unterschiedlichen Items in einer Regel $\xi$, dann ergibt sich die Anzahl der unterschiedlichen Komponenten $comp(\xi)$ wie folgt:[5]

$$\begin{aligned} |comp(\xi)| &= \sum_{k=1}^{n-1} \frac{n!}{k!(n-k)!} \\ &= \sum_{k=1}^{n-1} \binom{n}{k} \end{aligned}$$

Zusätzlich wächst die Menge der zu überprüfenden Komponenten linear mit der Anzahl interessanter Änderungen. Allerdings müssen nicht alle Komponenten eines Musters überprüft werden. Es wird angestrebt, alle interessanten Änderungen zu ermitteln, die zur Änderung der ursprünglichen Regel *beigetragen* haben. Dazu müssen zunächst nur die längsten Komponenten auf interessante Änderungen überprüft werden. Deren Anzahl ist jedoch mit

$$|comp_{longest}(\xi)| = \frac{n!}{(n-1)!}$$

weitaus kleiner als die Anzahl aller Komponenten. Erst wenn eine der Komponenten ebenfalls interessante Änderungen zeigt, muß diese weiter zerlegt werden. Es kann natürlich vorkommen, daß eine Komponente im Gegensatz zu ihren Elementen keine interessante Änderung zeigt. So ist es denkbar, daß der Support der Komponente $AB$ einer Regel $AB \Rightarrow C$ konstant bleibt, während sich die Support-Werte von $A$ und $B$ stark geändert haben. In diesem Fall wäre die Änderung der Regel *nicht* auf die untersuchte Komponente $AB$ zurückzuführen, weswegen die Änderungen ihrer Subkomponenten $A$ und $B$ nicht von Interesse sind und unbeachtet bleiben. Aus Effizienzerwägungen wird außerdem ein globaler Zwischenspeicher verwendet, mit dessen Hilfe sichergestellt ist, daß Komponenten nicht wiederholt geprüft werden.

In gewisser Weise ist dieser Ansatz mit dem Vorschlag von Liu u. a. [2001a] vergleichbar. In dieser Arbeit wird eine Methode zur Ermittlung der *fundamentalen* Regeländerungen entwickelt (vgl. Abschnitt 3.3.2). Es gibt jedoch einige grundlegende Unterschiede, die relativ starke Auswirkungen auf die

[5] Da die ursprüngliche Regel bereits betrachtet wurde, brauchen nur *echte* Untermengen von $\xi$ überprüft zu werden.

Performanz der vorgeschlagenen Technik haben. Ziel der Arbeit von Liu u. a. ist die Minimierung der Anzahl der zu meldenden Musteränderungen. Dies zwingt sie, *alle* gefundenen Muster in ihre Analyse einzubeziehen, was im Normalfall mit einem hohen Rechenaufwand verbunden ist, da die Komplexität auch bei diesem Algorithmus mit wachsender Regellänge stark ansteigt. Ein anderer wichtiger Unterschied ist die Einbeziehung von lediglich zwei Perioden – die Ermittlung langfristiger Änderungen ist damit grundsätzlich ausgeschlossen. Zusätzlich werden lediglich die signifikanten Änderungen zur Bestimmung fundamentaler Änderungen verwendet. Da langfristige Änderungen sich jedoch oft durch viele kleine, nicht notwendigerweise auch signifikante Änderungen manifestieren, ist die Entdeckung langfristiger Änderungen selbst bei Einbeziehung einer ausreichenden Anzahl von Perioden nur eingeschränkt möglich. Die wesentliche Kritik an diesem Ansatz betrifft jedoch die Kriterien, nach denen fundamentale Änderungen ermittelt werden. So wird die Änderung des Supports einer Regel $A \Rightarrow B$ als erklärbar und damit nicht fundamental klassifiziert, wenn die Support-Werte der Itemsets $A$ und $B$ die gleichen Änderungen aufweisen. Dies würde dazu führen, daß dem Nutzer nur die Änderungen für $A$ und $B$ angezeigt würden, was in jedem Fall einen Informationsverlust bedeutet. Gerade im Kontext einer Warenkorbanalyse ist die Information, daß die Produkte $A$ und $B$ häufiger zusammen verkauft werden, für den Anwender von besonderem Interesse, da er sich sonst darauf beschränken könnte, die häufigen Itemsets zu bestimmen. Zur Verdeutlichung der Problematik sei die Umkehrung des obigen Beispiels angeführt, wonach der Support der Komponente $AB$ stark ansteigen könnte, während die Support-Werte von $A$ und $B$ konstant bleiben.

## 5.6 Zusammenfassung

Bei der Erkennung von Regeländerungen sind zwei verschiedene Arten von Änderungen zu berücksichtigen – Änderungen des Inhalts der Muster und Änderungen ihrer statistischen Eigenschaften. Änderungen des Inhalts sind relativ schwer zu erkennen, da sich das Problem ihrer eindeutigen Identifizierung stellt. Die im Abschnitt 5.2.1 gegebenen Definitionen unterscheiden deshalb zwischen inhaltlichen Änderungen, die die Kennung eines Musters beeinflussen, und solchen, die keine Auswirkungen darauf haben. Die Erkennung von Änderungen der Statistiken eines Musters gestaltet sich dagegen deutlich einfacher, da sie bei Verwendung des im Kapitel 4 entwickelten Re-

gelmodells den Historien der statistischen Eigenschaften direkt entnommen werden können. Beeinflussen derartige Änderungen die Sichtbarkeit eines Musters, so sind auch ihre Auswirkungen auf der Regelbasis, der Gesamtheit aller gespeicherten Regeln, zu bestimmen. Durch die im Abschnitt 5.2.2 vorgenommene Gruppierung der Regelbasis erhält der Anwender in jeder Periode eine Vorstellung davon, in welchem Umfang die gefundenen Muster stabile oder nur temporär in Erscheinung tretende Zusammenhänge in den Daten widerspiegeln.

Im Hinblick auf die Auswahl interessanter Muster spielen konventionelle Maßstäbe bisher eine deutlich größere Rolle. Wie jedoch im Abschnitt 5.3.1 gezeigt wurde, ist diese Sichtweise mit vielen Problemen behaftet und für die Bewertung von Musteränderungen ungeeignet. So ist es vorstellbar, daß eine Regel, die den Kriterien in der Mining-Anfrage in allen Perioden genügte, keine oder nur unwesentliche Änderungen aufweist. Abhilfe verspricht die Einbeziehung der temporalen Eigenschaften der Muster. So können z. B. Ober- oder Untergrenzen für die absoluten und/oder relativen Änderungen benutzt werden, um interessante Änderungen zu bestimmen. Daneben ist die Verwendung statistischer Tests ein probates Mittel, um die Signifikanz von Änderungen sicherzustellen. Die Auswahl der interessierenden Änderungen kann jedoch auch nach subjektiven Kriterien erfolgen. Bei dieser Herangehensweise wählt der Nutzer die Muster entsprechend dem jeweiligen Anwendungskontext aus, um dann das Ausmaß der Änderungen zu definieren, die er für interessant hält. Nach diesem Ansatz wären nur jene Änderungen von Belang, die eines der ausgewählten Muster betreffen *und* die festgelegte Stärke aufweisen. Während dieser Ansatz sicher die besten Resultate liefert, ist er offensichtlich mit dem größten Aufwand verbunden.

Die im Abschnitt 5.3.4 vorgestellten Heuristiken berücksichtigen für die Bestimmung interessanter Änderungen verschiedene Aspekte der *Kontinuität* der Entwicklung von Mustern. Dabei betrachten sie Änderungen einzelner Muster und ihre Auswirkungen auf die Regelbasis. Die stabilitätsbasierte Heuristik fußt auf den im Abschnitt 5.2.2 definierten Regelgruppen und prüft, ob sich die Frequenz, mit der ein Muster gefunden werden kann, im Zeitverlauf ändert. Die korridorbasierte Heuristik hingegen bezieht die in der Vergangenheit gesammelten Erfahrungen hinsichtlich der Musterstatistiken ein und prüft in jeder Periode, ob der Wert der betrachteten Maßzahl unerwartet stark vom sonst üblichen Niveau abweicht. Die intervallbasierte Heuristik schließlich teilt den Wertebereich einer Statistik in Abschnitte gleicher Größe ein und sucht nach Verletzungen der Grenzen.

Nur in sehr seltenen Fällen wird direkt aus dem Datensatz ersichtlich, welche Ereignisse die Musteränderungen verursacht haben. Hinweise darauf kann zum einen die Betrachtung des zeitlichen Horizonts der beobachteten Änderungen liefern. Der im Abschnitt 5.4 vorgeschlagene Ansatz verwendet einen vom Nutzer vorgegebenen Parameter, um zu bestimmen, ob eine Änderung kurz- oder langfristiger Natur ist. Ein Vorteil der beschriebenen Methode liegt darin, daß sie nicht auf eine große Anzahl Perioden angewiesen ist, um den Zeithorizont einer Änderung abschätzen zu können. Zum anderen müssen jedoch auch jene Ursachen ermittelt werden, die unmittelbar aus dem untersuchten Datensatz hervorgehen. Zu diesem Zweck ermittelt der im Abschnitt 5.5 dargestellte Algorithmus, ob die Änderung eines Musters auf Änderungen seiner Bestandteile zurückgeführt werden kann. Je spezifischer die gewonnenen Erkenntnisse sind, desto wahrscheinlicher lassen sich Änderungen in den Daten auf ihre auslösenden Faktoren zurückführen.

# Kapitel 6

# Der PAM-Prototyp

## 6.1 Allgemeiner Aufbau einer PAM-Instanz

Die zentrale Aufgabe eines Pattern Monitor liegt in der Überwachung des entdeckten Wissens, und die in den vorangegangenen Kapiteln entwickelten Konzepte bilden das Grundgerüst eines solchen Monitors. Auf der einen Seite bietet PAM die grundlegenden Funktionalitäten, um Daten zu analysieren und die Ergebnisse verwalten und pflegen zu können, auf der anderen Seite eröffnet PAM die Möglichkeit, die gespeicherten Ergebnisse zu analysieren, um interessante Änderungen in ihnen zu erkennen.

Der schematische Aufbau einer PAM-Instanz ist aus Abbildung 6.1 ersichtlich. Im Kern ist die eigentliche Funktionalität von PAM implementiert. Neben der Benutzerschnittstelle bietet PAM eine Anbindung an verschiedene Mining-Algorithmen. Zur Zeit wird Weka, eine freie Sammlung in Java implementierter Algorithmen, verwendet [Witten und Frank 1999]. Prinzipiell können jedoch jederzeit weitere Algorithmen angebunden werden. Daten können in PAM über eine JDBC-Schnittstelle oder über einfache Textdateien importiert werden. Für die Speicherung der Daten und Muster verwendet PAM eine relationale Datenbank, mit der ebenfalls über JDBC kommuniziert wird.

## 6.2 Die Komponenten von PAM

Auch wenn nicht auf alle Implementierungsdetails eingegangen werden soll, folgt in diesem Abschnitt eine kurze Beschreibung der Funktionen, die von

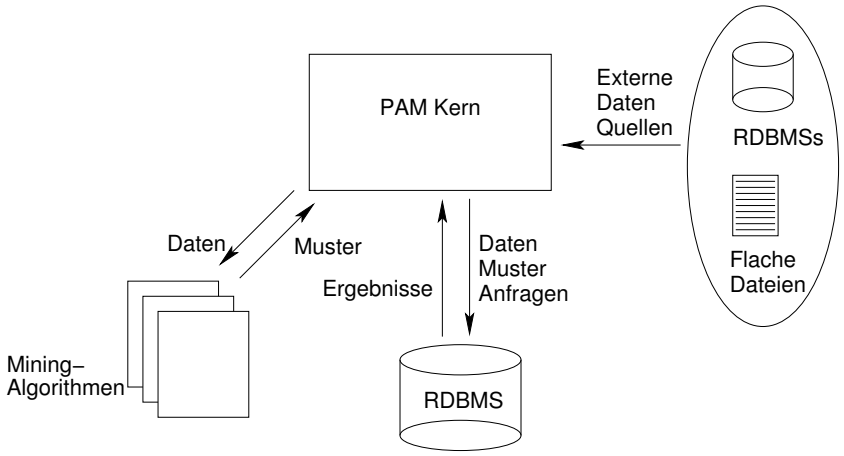


Abbildung 6.1: Allgemeiner Aufbau einer PAM-Instanz

den verschiedenen Komponenten wahrgenommen werden. Die Implementierung wurde im wesentlichen im Rahmen der beiden im nächsten Kapitel beschriebenen Fallstudien durchgeführt, wobei eine Vielzahl verschiedener Werkzeuge Verwendung fand. Die Auswahl richtete sich dabei nach den Anforderungen, die sich aus den Zielstellungen der geplanten Analysen ergaben, und unterstreicht den prototypischen Charakter der Implementierung.

### 6.2.1 Der Kern

Der Kern implementiert die Funktionen zur Erkennung und Analyse von interessanten Musteränderungen. Daneben enthält er eine Reihe von Hilfsklassen für die Nutzer- und Projektverwaltung. Die unterschiedlichen Teile des Kerns sind vollständig modularisiert und so generisch wie möglich gehalten, d. h., sie enthalten nur das jeweils notwendige Minimum an Anwendungslogik. Ein wesentlicher Vorteil der gewählten Vorgehensweise besteht darin, daß die Teile unabhängig voneinander und von einer bestimmten Nutzer-Schnittstelle verwendet werden können. Sie sind somit leicht in andere Anwendungen zu integrieren.

Die Grundlage bilden zwei Schnittstellen, die zum Datenaustausch mit anderen Anwendungen benutzt werden können. Durch ihre Nutzung ist es dem Anwender möglich, die zu importierenden Daten mit einem eindeutigen Zeitstempel zu versehen. Einerseits kann es sich dabei um Transaktionsdaten

handeln oder – sofern bereits gemäß dem temporalen Regelmodell organisiert – um die Mining-Ergebnisse. Die erste der beiden Schnittstellen erlaubt den Import von Daten aus einfachen ASCII-Dateien. Hierbei werden die Formate CSV (*Comma Separated Values*) und ARFF (*Attribute-Relation File Format*) unterstützt. Zum anderen kann ein Import von Daten aus JDBC-konformen Datenquellen erfolgen. Die Anbindung an die interne Datenhaltungskomponente von PAM ist ebenfalls über JDBC realisiert. Die Bestimmung der zu überwachenden Muster erfolgt entweder durch die in Form externer Bibliotheken vorliegenden Mining-Algorithmen, die über entsprechende Adapter-Klassen angesprochen werden, oder mit Hilfe der im Abschnitt 4.3.2 beschriebenen SQL-basierten Methode. Zusätzlich wird der Nutzer bei der Auswahl der zu überwachenden Muster unterstützt.

Die originäre Funktionalität des Kerns besteht jedoch in der Erkennung interessanter Änderungen und – in Abhängigkeit von den verwendeten Parametern – der Ermittlung der temporalen Dimension der beobachteten Änderungen, wobei auch deren Ursachen bestimmt werden können. Diese Funktionen sind jeweils als separate Klassen implementiert, wobei die Algorithmen als statische Methode realisiert sind. Zusätzlich verfügt jede Klasse über ein eigenes Hauptprogramm. Auf diese Weise können die Algorithmen über entsprechende Methodenaufrufe einfach in externe Anwendungen eingebunden, jedoch auch über eine einfache Kommandozeilen-Schnittstelle genutzt werden.

Im Mittelpunkt der Implementierung stehen die beiden Klassen `Pattern` und `Item`, welche die grundlegenden Abstraktionen für Muster und ihre Bestandteile darstellen. Die Schnittstellen dieser Klassen erlauben dem Anwender, die statistischen Eigenschaften eines Musters abzufragen. Um eine Instanz der Klasse `Pattern` zu erzeugen, reicht es aus, die Regelkennung und die gewünschte Periode anzugeben.[1] Der Konstruktor bestimmt alle anderen benötigten Informationen direkt aus den in der Datenbank gespeicherten Transaktionen. Außerdem verfügt die Klasse über die zwei Methoden `getPrevious()` und `getNext()`, die das Muster der Vorperiode bzw. der nächsten Periode zurückliefern und verwendet werden, um die Zeitreihen der Musterstatistiken ohne Miner zu bestimmen. Die Methode `store()` wird schließlich verwendet, um das aktuelle Objekt in der Datenbank zu speichern.

[1] Wie im Abschnitt 4.2.1 beschrieben wurde, bestimmt eine Funktion, die *Body* und *Head* der Regel als Eingabe verwendet, eine eindeutige Kennung für jede Regel. Umgekehrt kann diese Kennung natürlich auch verwendet werden, um den Inhalt der Regel festzulegen.

Unter Verwendung der beschriebenen Mechanismen greifen die Heuristiken dann auf die Datenquelle zu, um die Menge von Musteränderungen zu bestimmen, die analysiert werden sollen. Um die Persistenz der Ergebnisse zu gewährleisten, werden diese anschließend ebenfalls in der Datenbank erfaßt.

### 6.2.2 Datenhaltung

Die Funktion der Datenhaltungskomponente besteht in der Speicherung der Daten, Muster und Ergebnisse der verschiedenen Auswertungsschritte und wird mit Hilfe des relationalen Datenbanksystems Oracle8*i* realisiert.[2] Da die Verbindung zwischen dem Kern und der Datenbank unter Verwendung von JDBC hergestellt wird, kann im Prinzip jedoch auch jede andere Datenbank, für die ein entsprechender Treiber verfügbar ist, eingesetzt werden.

Die Speicherung der Muster in der Datenbank erfolgt unter Verwendung des im Abschnitt 4.2.2 dargestellten Relationenschemas, wobei anstelle einer generischen Spalte `statistics` weitere Spalten für Support, Konfidenz, Lift und Certainty Factor aufgenommen wurden. Außerdem enthält das Schema weitere Tabellen und Sichten, die für die Speicherung und kompakte Darstellung der Ergebnisse der Heuristiken und der Bestimmung der Ursachen von Musteränderungen verwendet werden. So gibt es unter anderem eine Sicht `rule_base_diffs`, welche die Änderungen der Regelbasis von einer Periode zur nächsten erfaßt.

### 6.2.3 Mining-Algorithmen

Die angeschlossenen Algorithmen dienen der Entdeckung einer initialen Menge von Mustern und – sofern sie nicht SQL-basiert ermittelt werden – der Bestimmung der Musterstatistiken. Obwohl PAM eigentlich als reiner Monitor konzipiert war, zeigte sich sehr schnell die Notwendigkeit, zumindest einige grundlegende Mining-Algorithmen zur Verfügung zu stellen. Ursprünglich konnte der Nutzer lediglich Mining-Ergebnisse importieren, die dann periodisch fortgeschrieben und überwacht werden sollten. Inzwischen wurde dieser Ansatz aber zugunsten einer Lösung aufgegeben, die den gesamten Prozeß der Wissensentdeckung einschließt. Das bedeutet, daß der Nutzer nur noch seine Daten importieren muß. Alle folgenden Schritte – von der

[2] `http://www.oracle.com` (24.02.2004)

Musterentdeckung, über die Verwaltung der Ergebnisse, bis hin zur Pflege und Analyse der Muster – sind direkt in PAM integriert.

Zur Zeit wird ein Teil der Algorithmen verwendet, die im Rahmen des Weka-Projekts (*Waikato Environment for Knowledge Analysis*) an der Universität von Waikato in Neuseeland implementiert wurden [Witten und Frank 1999]. Es handelt sich dabei um ein System, das viele verschiedene Methoden aus dem Bereich des Maschinellen Lernens zur Verfügung stellt, wie z. B. Cluster-Analyse und Methoden zur Klassifikation. Außerdem werden Funktionen für die Datenvorbereitung und die Visualisierung und Auswertung der Ergebnisse angeboten. Die Algorithmen sind in Java implementiert und unter der *GNU General Public License* veröffentlicht.[3]

Für die Warenkorbanalyse findet in PAM zur Zeit *Apriori* Verwendung und für die Cluster-Analyse *k-means* (vgl. Abschnitt 2.2). Für den Einsatz dieser Algorithmen wurden zwei Adapter-Klassen implementiert, welche die Aufrufe der Schnittstelle des verwendeten Algorithmus kapseln und dafür sorgen, daß die Mining-Ergebnisse direkt in der Datenbank erfaßt werden können. Auf diese Weise läßt sich prinzipiell auch jedes andere in Java implementierte Mining-Verfahren in PAM einbinden. Unter Verwendung des *Java Native Interface* (JNI) könnten jedoch auch weitere, nicht in Java implementierte Algorithmen einbezogen werden.

### 6.2.4 Die Nutzer-Schnittstelle

Wie bereits erwähnt, entstand der Monitor zu großen Teilen im Rahmen der durchgeführten Fallstudien. Da in diesem Zusammenhang keine Notwendigkeit für eine spezielle Nutzer-Schnittstelle bestand, wurde eine einfache Variante umgesetzt, die es gestattet, die für die verschiedenen Analyseschritte notwendigen Parameter als Kommandozeilenargumente anzugeben. Um die Arbeit mit PAM komfortabler zu gestalten, wurde außerdem eine Menüsteuerung für diese Lösung entworfen. Daneben befindet sich zur Zeit eine WWW-basierte Lösung in der Entwicklung, welche die verschiedenen, im nächsten Abschnitt beschriebenen Vorgehensweisen zur Pflege und Auswertung der Analyse-Ergebnisse umsetzt. Kern der neuen Schnittstelle ist der Servlet-Container *Tomcat*, in dem PAM ausgeführt wird und der es erlaubt, die existierende Implementierung weiterhin zu nutzen.[4]

[3] `http://www.gnu.org` (24.02.2004)

[4] `http://jakarta.apache.org/tomcat` (24.02.2004)

## 6.3 PAM-Workflows

Abbildung 6.2 zeigt, wie die Funktionen von PAM, die durch Rechtecke dargestellt sind, interagieren und welche Ein- und Ausgaben sie haben. Der Im-

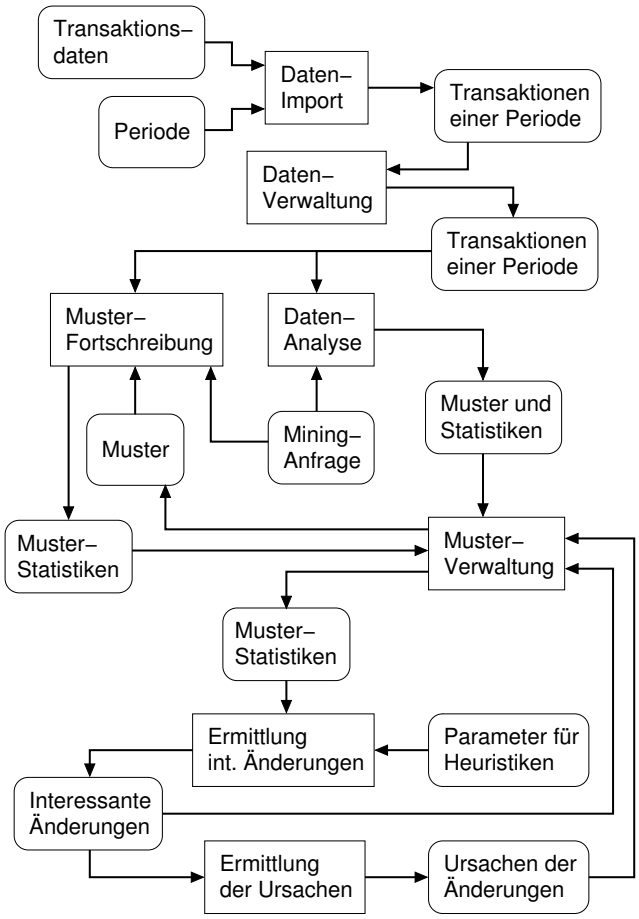


Abbildung 6.2: Interaktion der PAM-Funktionen

port dient der Periodisierung der Transaktionsdaten, welche in Verbindung mit der Mining-Anfrage als Basis für die Analyse und/oder die Fortschreibung der Statistiken einer vorgegebenen Menge von Mustern fungieren. Die resultierenden Zeitreihen werden auf interessante Änderungen hin untersucht, welche dann als Eingabe für die Ermittlung der Ursachen der beobachteten Änderungen verwendet werden.

PAM kann unter Verwendung vordefinierter *Workflows* genutzt werden, die sich bezüglich der Verzahnung der reinen Fortschreibung einmal entdeckter Muster und der neuerlichen Analyse der Daten unterscheiden. Die Beschreibung der Workflows soll die Interaktion der Komponenten und ihrer jeweiligen Funktionen verdeutlichen.

### 6.3.1 Permanentes Mining

Dieser Modus stellt gewissermaßen den Standard-Anwendungsfall von PAM dar, bei dem der Monitor *ausschließlich* zur Erkennung interessanter Änderungen verwendet wird. Dabei werden die Daten periodisch importiert und mit Hilfe der integrierten Mining-Software analysiert. Die Ergebnisse der Analyse werden in der angeschlossenen Datenbank abgelegt, d. h., neue Muster werden eingetragen und die Statistiken bereits bekannter Muster fortgeschrieben. Zu jedem Zeitpunkt können dann die verschiedenen Heuristiken benutzt werden, um interessante Änderungen für *alle* gespeicherten Muster zu erkennen. Im Anschluß kann der Anwender dann die Ursachen für die beobachteten Musteränderungen bestimmen lassen.

Ein Nachteil dieser Methode ist, daß die Statistiken von Mustern, die bereits in der Regelbasis enthalten sind, jedoch in einer der folgenden Perioden den vorgegebenen Mining-Parametern nicht genügt haben, nicht fortgeschrieben werden könnten. Dadurch würden die Zeitreihen der statistischen Eigenschaften unterbrochen, und Standardverfahren wie Zeitreihenanalysen könnten nicht mehr angewandt werden. Erschwerend kommt hinzu, daß auch einige der Heuristiken auf die Vollständigkeit der Zeitreihen angewiesen sind. Außerdem ist in jeder Periode ein unter Umständen nicht zu vernachlässigender Rechenaufwand erforderlich, um die Analyse durchzuführen.

### 6.3.2 Periodisches Mining

Unter Verwendung dieses Ansatzes werden die importierten Daten nicht mehr in jeder Periode analysiert, sondern nur noch in gewissen Abständen. Jede Mining-Session liefert eine Menge Muster, die in den Folgeperioden, also bis zur nächsten Mining-Session, fortgeschrieben werden (vgl. Abschnitt 4.3.2). Durch eine erneute Mining-Session wird dann die Regelbasis aktualisiert, und die Muster werden bis zur nächsten Session weiter fortgeschrieben. Bei jedem Aufruf des Miners kann der Fall eintreten, daß neue Muster entdeckt werden

oder alte verschwinden. Neue Muster können ohne weiteres in die Regelbasis aufgenommen werden. Bei Mustern, die nicht mehr in den Ergebnissen erscheinen, muß entschieden werden, ob ihre Statistiken weiter fortgeschrieben werden sollen oder sie aus der Regelbasis entfernt werden sollen. Diese Entscheidung hängt im wesentlichen von der *Stabilität* des betrachteten Datensatzes ab. Ist der Datensatz im Zeitverlauf starken Änderungen ausgesetzt, kann es viele temporäre Muster geben, die immer nur für kurze Zeit präsent sind. In diesem Fall wäre die Fortschreibung aller Muster mit einem stetig steigenden Aufwand verbunden, da periodisch immer wieder neue Muster hinzukämen, jedoch niemals Muster aus der Regelbasis entfernt würden. Außerdem ist in einem solchen Fall die Aktualität der Regelbasis stark beeinträchtigt, weswegen unter Abwägung der Vor- und Nachteile eventuell sogar das permanente Mining vorzuziehen ist. Des weiteren darf natürlich der Anwendungskontext nicht unbeachtet bleiben. Werden Muster aus der Regelbasis entfernt, die eventuell nur in einer Periode abwesend sind, verfälscht dies das Gesamtbild der Analyse mindestens bis zur nächsten Mining-Session. Ob dieser Mangel in Kauf zu nehmen ist, muß in Abhängigkeit von der jeweiligen Anwendung entschieden werden.

Der Abstand zwischen den Mining-Sessions kann nach verschiedenen Kriterien bestimmt werden. Zum einen kann der Zeitraum zwischen zwei aufeinanderfolgenden Sessions fest vorgegeben werden, zum anderen kann die Länge des Intervalls in Abhängigkeit von den ermittelten Änderungen dynamisch festgelegt werden. So könnte der Nutzer Grenzen für absolute und/oder relative Abweichungen für eine oder mehrere statistische Eigenschaften der Muster vorgeben, bei deren Über- bzw. Unterschreitung dem Nutzer die Notwendigkeit einer weiteren Mining-Session signalisiert würde. Nach diesem Prinzip wird auch die Pflege der überwachten Cluster vorgenommen. Solange das Clustering *gültig* ist, werden die Daten der Folgeperioden gemäß der vorhandenen Cluster klassifiziert. Ist das aktuelle Clustering ungültig, werden auf der Grundlage der Daten der aktuellen Periode neue Cluster abgeleitet (vgl. Abschnitt 4.3.2).

Unabhängig davon, ob die Änderungen im Rahmen einer Mining-Session oder durch die Fortschreibung der Muster bestimmt wurden, können diese mit Hilfe der durch den Kern bereitgestellten Werkzeuge in jeder Periode analysiert werden, um die interessanten Änderungen und ihre Ursachen zu bestimmen.

### 6.3.3 Überwachung interessanter Muster

Dieser Ansatz kann als Spezialfall der periodischen Analyse der Daten betrachtet werden, obwohl eigentlich kein Miner mehr verwendet werden muß. In Abhängigkeit vom Anwendungskontext definiert der Nutzer eine Reihe von *Templates*, um die Muster zu beschreiben, an denen er interessiert ist. PAM extrahiert dann in jeder Periode die Statistiken aller Muster, die diesem Template genügen. Im Rahmen der Auswertung der festgestellten Änderungen werden dem Nutzer dann alle interessanten Veränderungen, die diese Muster betreffen, gemeldet und ihre Ursachen bestimmt.

Ein solches Vorgehen ist immer dann von Interesse, wenn Form und Inhalt interessanter Muster bereits vor Beginn der Analyse bekannt sind. Als Beispiel sei eine Sicherheitsanwendung angeführt, in deren Rahmen periodisch nach vordefinierten Fehlern gesucht werden soll. Ein weiteres Beispiel ist die im Rahmen der Fallstudien beschriebene Analyse von Kommunikationsmustern. Dabei wurden einige Muster entdeckt, welche die erlaubten Operationen des Mail-Servers darstellten und hohe Konfidenz-Werte von fast 100% aufwiesen. Um die korrekte Nutzung des Mail-Servers zu überprüfen, müßte der Administrator also lediglich untersuchen, ob die Muster vorhanden sind und ob sie hohe Konfidenzen aufweisen (vgl. Beispiel 5.3).

Eine Kombination der periodischen Analyse und der reinen Fortschreibung der statistischen Eigenschaften interessanter Muster wurde in [Baron u. a. 2003] vorgestellt. Die zu beobachtenden Muster wurden dabei nicht im vorhinein angegeben, sondern in der ersten Periode mit Hilfe des Miners ermittelt. In den Folgeperioden sollten dann die Statistiken dieser Muster fortgeschrieben werden, *solange* sie die in der Mining-Anfrage angegebenen Grenzen für Support und Konfidenz überschritten. Auf diese Weise wurden mit steigender Periodenzahl nur noch Muster beobachtet, die permanent, d. h. in jeder Periode in den Daten gefunden werden konnten. Dieser Vorgehensweise lag die Idee zugrunde, nur solche Muster zu beobachten, die den invarianten Teil der Population repräsentieren, und Änderungen der Population indirekt über die Änderungen, denen die permanenten Muster ausgesetzt waren, zu erkennen. Demnach wurde eine neue Mining-Session immer dann durchgeführt, wenn die Regelbasis starke Änderungen aufwies.

Die Vorteile der beschriebenen Methode liegen auf der Hand. Die ungerichtete Suche des Miners nach Mustern wird durch die gezielte Bestimmung der Statistiken der interessanten Muster ersetzt (vgl. Abschnitt 4.3.2). Im Vergleich mit der konventionellen Musterentdeckung ist die vorgeschlagene

Methode deswegen deutlich effizienter, birgt jedoch den Nachteil in sich, daß auf diese Weise keine neuen Muster entdeckt werden können.

### 6.3.4 Kumulative Überwachung aller Muster

Dieser Ansatz liefert eine vollständige Übersicht über *alle* Muster, die im Verlauf der bisherigen Analyse entdeckt wurden. Dazu wird in jeder Periode eine Mining-Session durchgeführt, um die in der aktuellen Partition der Daten enthaltenen Muster zu bestimmen. Alle gefundenen Muster werden in der Datenbank gespeichert, d. h., die statistischen Eigenschaften bekannter Muster werden fortgeschrieben, neue Muster werden eingetragen. In einem zweiten Schritt werden die statistischen Eigenschaften der neuen Muster für alle bisherigen Perioden ermittelt und nachträglich in der Datenbank erfaßt. Im dritten Schritt werden dann zusätzlich die Statistiken jener Muster, die bereits bekannt sind, jedoch in der aktuellen Periode nicht gefunden werden konnten, aus den zugrunde liegenden Basisdaten ermittelt und in der Datenbank eingetragen. Anschließend werden dann alle beobachteten Änderungen auf ihre Wichtigkeit hin untersucht und – bei Bedarf – deren Ursachen ermittelt.

Der Vorteil dieser Methode ist, daß zu jedem Zeitpunkt die vollständigen Zeitreihen aller bisher gefundenen Muster vorliegen. Damit können sowohl die Heuristiken zur Bestimmung interessanter Musteränderungen verwendet werden als auch Standardverfahren der Zeitreihenanalyse. Selbst für jene Muster, die in der aktuellen Periode zum ersten Mal entdeckt wurden, ist die bisherige Evolution für den Nutzer vollständig nachvollziehbar. Der Preis für diese zusätzlichen Informationen wird am Aufwand ersichtlich, in jeder Periode eine Mining-Session durchzuführen und für eine potentiell kontinuierlich wachsende Menge an Mustern die statistischen Eigenschaften zu bestimmen. Andererseits ist der Bearbeitungsaufwand immer noch deutlich kleiner als bei der unbeschränkten Mustersuche, da in diesem Fall alle im Datensatz enthaltenen Muster gesucht werden müssen [Baron und Spiliopoulou 2001].

## 6.4 Zusammenfassung

Die zentrale Aufgabe von PAM liegt in der Verwaltung und Analyse des entdeckten Wissens. Dabei unterstützt PAM jedoch den *vollständigen* Prozeß der Wissensentdeckung – von der Datenanalyse, über die Pflege der entdeck-

ten Regeln, bis hin zur Untersuchung der beobachteten Änderungen in den gefundenen Ergebnissen. Der Monitor definiert eine Reihe von Workflows, die sich bezüglich der Verzahnung von regulärer Datenanalyse mit Hilfe konventioneller Mining-Techniken und datenbankgestützter Fortschreibung des entdeckten Wissens unterscheiden und deren Unterschiede und Gemeinsamkeiten in Tabelle 6.1 zusammengefaßt sind. Dazu verfügt PAM über eine

| | Permanentes Mining | Periodisches Mining | Überwachung interessanter Muster | Kumulative Überwachung aller Muster |
|---|---|---|---|---|
| Mining-Session | in jeder Periode | periodisch | eventuell in Periode $t_0$ | in jeder Periode |
| Fort-schreibung | – | in jeder Periode | in jeder Periode ohne Mining-Session | in jeder Periode |
| Ermittlung interessanter Änderungen | für alle beobachteten Änderungen | für alle beobachteten Änderungen | für alle beobachteten Änderungen überwachter Muster | für alle beobachteten Änderungen |
| Ursachen-ermittlung | für alle interessanten Änderungen | für alle interessanten Änderungen | für alle interessanten Änderungen überwachter Muster | für alle interessanten Änderungen |

Tabelle 6.1: Gegenüberstellung der unterschiedlichen PAM-Workflows

Reihe von Schnittstellen, die den Import von Daten aus anderen Anwendungen gestatten. Zur Musterentdeckung verwendet PAM externe Mining-Bibliotheken, die sich über einen einfachen Adapter-Mechanismus integrieren lassen. Die interne Datenhaltung realisiert PAM mit Hilfe eines relationalen Datenbanksystems, in dem jedoch nicht nur die der Analyse zugrunde liegenden Basisdaten abgelegt sind, sondern auch alle Mining-Ergebnisse, die gemäß dem temporalen Regelmodell organisiert sind. Der Kern von PAM implementiert die verschiedenen Verfahren zur Analyse der Evolution des

entdeckten Wissens, deren Ergebnisse ebenfalls in der Datenbank erfaßt werden.

Als Resultat liegen dem Anwender nicht nur die in den verschiedenen Perioden gefundenen Muster vor, sondern auch die Historien ihrer statistischen Eigenschaften, für welche die gemäß den Vorgaben des Nutzers für interessant befundenen Abweichungen gemeldet werden. Die interessanten Änderungen betreffend wird dem Nutzer darüber hinaus signalisiert, welche Änderungen in den Daten als Ursache in Frage kommen bzw. dazu beigetragen haben.

# Kapitel 7

# Experimentelle Ergebnisse

Um die vorgestellten Verfahren zur Pflege des entdeckten Wissens und zur Analyse seiner Evolution zu prüfen, wurden zwei Fallstudien durchgeführt, die in den folgenden Abschnitten beschrieben sind. In beiden Fallstudien wurden Transaktionsdaten, die über einen längeren Zeitraum gesammelt wurden, mit Hilfe einer Assoziationsanalyse untersucht. Während der Fokus der ersten Fallstudie vorrangig auf der SQL-basierten Fortschreibung der entdeckten Muster lag, wurden in der zweiten Fallstudie die Heuristiken zur Entdeckung interessanter Musteränderungen überprüft.

## 7.1 Beobachtung der Änderungen von Kommunikationsmustern

Ziel dieser Fallstudie war die Klärung der Frage, ob die SQL-basierte Überwachung der gefundenen Muster die aufwendige Suche nach Mustern mit Hilfe einer Mining-Software in jeder Periode ersetzen kann [Baron u. a. 2003]. Im Rahmen dieser Untersuchung wurde das Transaktionsprotokoll eines Mail-Servers analysiert, der am Institut für Wirtschaftsinformatik der Humboldt-Universität zu Berlin betrieben wird. Die Protokolldatei enthielt insgesamt rund 280 000 Einträge, die von etwa 103 000 gesendeten Nachrichten stammen und im Verlauf eines Jahres gesammelt wurden. Abbildung 7.1 zeigt einen Ausschnitt des Protokolls, wie es von *Sendmail*, einem weitverbreiteten SMTP-Server, angelegt wird.[1]

[1] `http://www.sendmail.org` (24.02.2004)

```
2000 Jan  1 00:00:01 mailhost sendmail[1877]: AAA01877:
 from=sender@sendhost, size=1111, class=0, pri=31111, nrcpts=1,
 msgid=199912312300.AAA00605@sendhost, proto=SMTP, relay=sendhost
2000 Jan  1 00:00:02 mailhost sendmail[1887]: AAA01877:
 to=recipient@mailhost, ctladdr=sender@sendhost, delay=00:00:01,
 mailer=local, stat=Sent
```

Abbildung 7.1: Auszug aus dem Transaktionsprotokoll von Sendmail

Für jede Nachricht, die von Sendmail zugestellt wird, erscheinen im Transaktionsprotokoll $n + 1$ Einträge, wobei $n$ der Anzahl unterschiedlicher Empfänger der Nachricht entspricht. Die Nachricht, die im angegebenen Beispiel protokolliert wurde, hat nur einen Empfänger, weswegen zwei Einträge im Protokoll erscheinen. Der erste Eintrag protokolliert den Eingang der Nachricht beim Mail-Server (`from`-Zeile). Neben dem Datum und der Zeit des Eingangs wird der Name des Mail-Servers (`mailhost`) und die Nummer des empfangenden Sendmail-Prozesses im Betriebssystem (`1877`) vermerkt. Zusätzlich wird eine alphanumerische Transaktionskennung erzeugt, die jedoch nur über einen Zeitraum von fünf Tagen eindeutig ist (`AAA01877`). Das Argument des `from`-Elements bezeichnet den Absender der Nachricht. Es folgen weitere Attribute, die die Größe der Nachricht in Bytes, die Klasse, die Priorität und die Anzahl der Empfänger der Nachricht angeben. Außerdem wird eine eindeutige Kennung für jede Nachricht erzeugt, das Protokoll, das für den Versand der Nachricht benutzt wurde, und der Name des Servers, von dem die Nachricht entgegengenommen wurde, gespeichert.

Der zweite Eintrag protokolliert die Zustellung der E-Mail an ihren Empfänger bzw. die Weiterleitung an den zuständigen Mail-Server (`to`-Zeile). Neben den Angaben, die auch schon in der entsprechenden `from`-Zeile erfaßt sind, protokolliert das Argument des `to`-Elements den Adressaten der Nachricht und das Argument des `ctladdr`-Eintrags den Absender. Zusätzlich gibt es zwei Attribute, von denen das eine die Zeitspanne zwischen Empfang und Zustellung und das andere den Status der Nachricht angibt.

### 7.1.1 Datenvorbereitung

Aus diesem Transaktionsprotokoll sollten Muster abgeleitet werden, welche die Kommunikation zwischen verschiedenen Nutzergruppen beschreiben.[2] Außerdem sollte untersucht werden, zu welchen Zeiten Nachrichten verschickt werden und wie viele Empfänger diese Nachrichten haben. In Abhängigkeit von der Herkunft wurden die in Tabelle 7.1 dargestellten Nutzergruppen festgelegt. Um aussagekräftigere Muster zu erhalten, wurde aus dem Datum

| Nutzergruppe | Zugehörigkeit |
|---|---|
| Institut | alle Mitarbeiter des Instituts |
| Fakultät | alle Mitarbeiter der Fakultät, die *nicht* zum Institut gehören |
| extern | alle Personen, die weder zum Institut noch zur Fakultät gehören |

Tabelle 7.1: Die Nutzergruppen des Mail-Servers

und der Uhrzeit ein kategorisches Attribut abgeleitet. Dazu wurde die Zeit, zu der eine Nachricht verschickt wurde, in Abhängigkeit von der Tageszeit, vom Wochentag und von der Frage, ob es sich um einen Feiertag handelt, in Arbeits- und Freizeit eingeteilt.[3] Alle Informationen wurden in einem neunstelligen *Multi*-Attribut kodiert, das zusätzlich die Anzahl der Empfänger einer Nachricht angibt (vgl. Tabelle 7.2).

**Beispiel 7.1** Entsprechend der Herkunft von Absender (`sender`) und Empfänger (`recipient`) könnte die in Abbildung 7.1 gezeigte Nachricht durch die in Tabelle 7.3 aufgeführten Kombinationen von `from`- und `to`-Items dargestellt werden. Die Nachricht wurde außerhalb der Arbeitszeit verschickt, weshalb die erste Stelle des Multi-Attributs in allen Fällen eine Null enthält. Während das Item, das einen `from`-Eintrag repräsentiert, an den Positionen 5 bis 7 eine Null aufweist, enthält ein `to`-Item an den Positionen 2 bis 4

[2] Auf eine Analyse der Kommunikation auf der Ebene einzelner Nutzer wurde aus praktischen Erwägungen verzichtet, da die resultierenden Muster einen äußerst geringen Support aufweisen würden. Zudem wäre eine solche Vorgehensweise datenschutzrechtlich äußerst bedenklich.

[3] Als Arbeitszeit wurde der Zeitraum von 8.00 bis 17.00 Uhr gewählt.

| Position | Beschreibung |
|---|---|
| 1 | Arbeitszeit(1) bzw. Freizeit (0) |
| 2 | Absender gehört zum Institut (1) oder nicht (0) |
| 3 | Absender gehört zur Fakultät (1) oder nicht (0) |
| 4 | Absender ist extern (1) oder nicht (0) |
| 5 | Empfänger gehört zum Institut (1) oder nicht (0) |
| 6 | Empfänger gehört zur Fakultät (1) oder nicht (0) |
| 7 | Empfänger ist extern (1) oder nicht (0) |
| 8<br>9 | Anzahl der Empfänger (Dezimalzahl) |

Tabelle 7.2: Kodierung für das Multi-Attribut

| from-Zeile | to-Zeile | Absender | Empfänger |
|---|---|---|---|
| 010000001 | 000010001 | Institut | Institut |
| 010000001 | 000001001 | Institut | Fakultät |
| 010000001 | 000000101 | Institut | extern |
| 001000001 | 000010001 | Fakultät | Institut |
| 001000001 | 000001001 | Fakultät | Fakultät |
| 001000001 | 000000101 | Fakultät | extern |
| 000100001 | 000010001 | extern | Institut |
| 000100001 | 000001001 | extern | Fakultät |
| 000100001 | 000000101 | extern | extern |

Tabelle 7.3: Beispiele für die Kodierung von Sender und Empfänger

eine Null. Für korrespondierende from- und to-Einträge ist die Anzahl der Empfänger jeweils gleich (1). ◇

Nach der Reinigung und Transformation in der oben beschriebenen Form wurde das Transaktionsprotokoll in ein relationales Datenbanksystem importiert. Zu diesem Zweck wurde das Protokoll auf Basis der Kalenderwoche geteilt und wurden die resultierenden Partitionen in separaten Tabellen gespeichert. Während die meisten Perioden zwischen 1700 und 2700 Transaktionen enthielten, lag das Minimum einer einzelnen Periode bei rund 270 und das Maximum bei ca. 5000 Transaktionen.

### 7.1.2 Datenanalyse

Im Mittelpunkt dieses Anwendungsfalls stand das Bestreben, die erhaltenen Ergebnisse möglichst effizient zu aktualisieren. Interessante Musteränderungen im Zeitverlauf sollten erkannt werden, ohne die Daten jeder Periode einer regulären Analyse durch eine Mining-Software unterziehen zu müssen. Dieses sollte erreicht werden, indem einerseits nur eine Untermenge der gefundenen Muster überwacht werden sollte und andererseits die Zeitreihen der statistischen Eigenschaften der beobachteten Muster direkt aus den der Analyse zugrunde liegenden Transaktionsdaten bestimmt würden. Dazu wurde für jedes zu beobachtende Muster eine Menge von SQL-Anfragen generiert, die diese Statistiken aus der Datenbank extrahierten (vgl. Abschnitt 4.3.2).

Die Analyse der Daten fand dann in einem zweistufigen Prozeß statt. In der ersten Phase, der Mining-Phase, wurden die Daten der ersten Periode einer Warenkorbanalyse unterzogen. In der zweiten, der Beobachtungsphase, die aus allen folgenden Perioden bestand, sollte ein Teil der gefundenen Muster dann aktualisiert und beobachtet werden, um interessante Änderungen zu entdecken. Starke Änderungen der beobachteten Muster dienen dabei als Indikator für grundlegende Änderungen im untersuchten Datensatz und signalisieren die Notwendigkeit einer neuen Mining-Session.

#### Mining-Phase

Für die initiale Musterentdeckung wurde in diesem Fall der SAS Enterprise Miner verwendet. Die Mining-Anfrage bestand dabei aus der Angabe von unteren Grenzen für den Support ($\tau_s = 0{,}05$) und die Konfidenz ($\tau_c = 0{,}1$).[4] Der verwendete Miner liefert grundsätzlich nur Muster zurück, die mindestens einen Lift von $l = 1$ aufweisen (vgl. Abschnitt 2.2). Da für die Untersuchung ein externer Miner verwendet wurde, mußten die Daten der ersten Periode zunächst wieder exportiert werden. Nachdem die Analyse abgeschlossen war, wurden die Ergebnisse unter Verwendung des in ein relationales Datenmodell transformierten temporalen Regelmodells wieder in der Datenbank gespeichert (vgl. Abschnitt 4.2.2).

Insgesamt wurden im Datensatz der ersten Periode 21 Muster gefunden, von denen ein Teil in Tabelle 7.4 dargestellt ist. Im oberen Teil der Tabelle sind Muster angegeben, die das allgemeine Kommunikationsverhal-

[4]Die Grenze für den Support bezieht sich in diesem Fall nicht auf die Gesamtzahl der Transaktionen, sondern auf die größte Einzelhäufigkeit eines Items.

| Nr. | Body | Head | Konfidenz | Support | Lift | Anzahl |
|---|---|---|---|---|---|---|
| 1. | 110000002 | 100010002 | 0,905 | 0,026 | 20,05 | 38 |
| 2. | 010000001 | 000010001 | 0,680 | 0,114 | 1,70 | 170 |
| 3. | 110000001 | 100010001 | 0,489 | 0,073 | 1,61 | 109 |
| 4. | 110000001 | 100000101 | 0,489 | 0,073 | 5,63 | 109 |
| 5. | 010000001 | 000000101 | 0,316 | 0,053 | 4,65 | 79 |
| 6. | 000100002 | 000010002 | 1,000 | 0,026 | 25,17 | 39 |
| 7. | 101000001 | 100010001 | 0,973 | 0,024 | 3,20 | 36 |
| 8. | 000100001 | 000010001 | 0,949 | 0,276 | 2,37 | 410 |
| 9. | 100100001 | 100010001 | 0,939 | 0,206 | 3,09 | 306 |
| 10. | 000000102 | 000010002 | 0,923 | 0,024 | 23,23 | 36 |

Tabelle 7.4: Auswahl der Ergebnisse der ersten Mining-Session

ten der Nutzer des Mail-Servers beschreiben. Das erste Muster repräsentiert z. B. Nachrichten, die während der Arbeitszeit von Mitgliedern des Instituts an zwei Empfänger geschickt wurden, von denen mindestens einer ebenfalls zum Institut gehört. Während das zweite und das dritte Muster den gleichen Zusammenhang für E-Mails an einen einzelnen Empfänger außerhalb und während der Arbeitszeit repräsentieren, stellen die Muster 4 und 5 diesen Zusammenhang für E-Mails an externe Adressaten dar. Es stellte sich heraus, daß zwischen den Mitgliedern des Instituts relativ viele Nachrichten ausgetauscht werden, jedoch kaum zwischen Mitgliedern des Instituts und der Fakultät. Wenn letzteres doch der Fall war, wurden die Nachrichten meist während der Arbeitszeit gesendet (Muster 7).

Im unteren Teil von Tabelle 7.4 sind Muster angegeben, die die *erlaubten* Operationen des Mail-Servers darstellen. Diese Muster repräsentieren ausnahmslos Nachrichten, die *nicht* von Mitgliedern des Instituts versandt wurden. Um sicherzustellen, daß der Mail-Server nicht unbefugt benutzt wird, sollten derartige Nachrichten immer auch an ein Mitglied des Instituts gerichtet sein (vgl. Beispiel 3.3). Darüber hinaus sollten diese Muster mit sehr hoher Konfidenz gelten. Muster 6 sagt z. B. aus, daß E-Mails an zwei Adressaten, die außerhalb der Arbeitszeit von einem externen Absender geschickt wurden, mindestens einen Empfänger haben, der zum Institut gehört. Dieses Muster gilt ohne Ausnahme, d. h. mit einer Konfidenz von 100%. Das letzte Muster stellt den gleichen Zusammenhang zwischen den beiden Empfängern einer E-Mail dar. Wenn eine Nachricht mit zwei Empfängern an einen ex-

ternen Adressaten geschickt wurde, ist der zweite Adressat ein Mitglied des Instituts. Dieses Muster gilt jedoch nicht mit einer Konfidenz von 100%. Wie sich im Zuge der Auswertung der Muster zeigte, ist die Ursache dafür nicht in einer unbefugten Nutzung des Mail-Servers zu sehen. Vielmehr liegt dies in der Tatsache begründet, daß nicht alle Nachrichten an zwei Adressaten auch einen externen Empfänger haben, da E-Mails natürlich auch an zwei interne Empfänger geschickt werden können. Die Muster 8 und 9 zeigen außerdem, daß fast die Hälfte aller Nachrichten von externen Nutzern an Mitglieder des Instituts geschickt werden, der größere Teil davon außerhalb der Arbeitszeit.

### Monitoring-Phase

Der erste Schritt in dieser Phase der Analyse bestand darin, die Muster auszuwählen, die in den folgenden Perioden überwacht werden sollten. Dieser Schritt ist im allgemeinen stark anwendungsabhängig, die Auswahl kann jedoch auch nach objektiven Kriterien vorgenommen werden (vgl. Abschnitt 3.3.1). In diesem Fall wurde kein konventioneller Ansatz verfolgt, sondern die Regeln wurden aufgrund ihrer *temporalen* Eigenschaften ausgewählt (vgl. Abschnitt 5.3). Im einzelnen sollten alle Regeln überwacht werden, die in allen Perioden in den Daten vertreten waren. Der Auswahl lag die Idee zugrunde, daß die Muster, die permanent in den Daten gefunden werden können, Teile der invarianten Eigenschaften des Datensatzes repräsentieren und ihre Überwachung deshalb wichtige Aufschlüsse über Änderungen in den untersuchten Daten liefern kann (vgl. Abschnitt 5.4.2).

Zu Beginn der Monitoring-Phase ist es natürlich nicht möglich zu entscheiden, welche Muster in jeder Periode in den Daten gefunden werden. Es ist jedoch klar, daß sie bereits in den Ergebnissen der Mining-Phase enthalten sind. Deswegen wurden zunächst alle Muster ausgewählt, die aus der ersten Phase stammten. In den folgenden Perioden wurde dann für jedes Muster geprüft, ob es die Grenzen für Support und Konfidenz überschritt.

Die Vorgehensweise bei der Auswahl der Muster war im betrachteten Fall unproblematisch, da nur eine relativ kleine Anzahl von Mustern gefunden wurde. In einer realen Anwendung ist es jedoch wahrscheinlich, daß die Anzahl der gefundenen Regeln sehr groß ist. Zu Beginn der Monitoring-Phase werden deswegen sehr viele Muster zu überwachen sein, was unter Umständen einen nicht unbeträchtlichen Aufwand darstellen kann. Es ist jedoch damit zu rechnen, daß die Anzahl der zu überwachenden Muster mit steigender Periodenzahl relativ schnell abnimmt. Ist der Aufwand trotzdem zu groß,

könnten die Muster zusätzlich nach statischen Kriterien gefiltert werden.

Im zweiten Schritt wurden dann für alle folgenden Perioden die statistischen Eigenschaften der beobachteten Muster bestimmt. Erfüllte ein Muster dabei nicht die in der Mining-Anfrage angegebenen Bedingungen, wurde es aus der Regelbasis entfernt. Abbildung 7.2 faßt die Ergebnisse der Analyse zusammen. Neben kleineren Änderungen in den Perioden 3 und 31 sind zwei

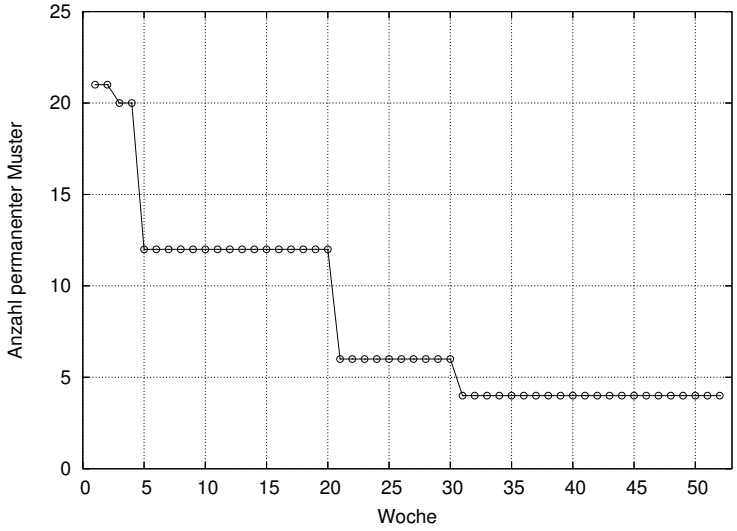


Abbildung 7.2: Entwicklung der Anzahl permanenter Muster

sehr starke Änderungen in den Perioden 5 und 21 zu beobachten. In der fünften Woche geht die Anzahl der Regeln, bezogen auf die Vorperiode, um 40% und in der 21. Woche um 50% zurück. Zudem stellten jeweils zwei der in Periode 31 verbliebenen vier permanenten Muster den gleichen Zusammenhang in den Daten dar, weswegen in beiden Fällen das Muster mit der niedrigeren Konfidenz entfernt wurde. Am Ende der Analyse konnten die invarianten Beziehungen innerhalb des Datensatzes durch gerade zwei Regeln beschrieben werden.

Abbildung 7.3 zeigt die Entwicklung des Supports dieser beiden Muster über den gesamten Analysezeitraum. Sowohl die Änderung in Periode 5 als auch jene in Periode 21 sind mit deutlichen Support-Änderungen des Musters M1 verbunden. Zusätzlich ist für beide Muster eine starke Änderung des Supports in der letzten Periode erkennbar, für die es keine korrespondierende

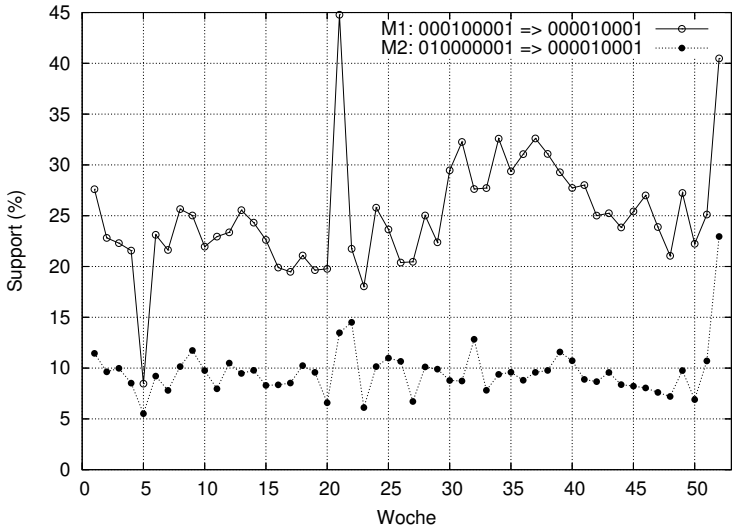


Abbildung 7.3: Support der permanenten Regeln

Änderung der Anzahl permanenter Regeln gibt. Es ist außerdem ersichtlich, daß die beiden Muster gemeinsam in Periode 21 fast 50% und in Periode 52 sogar fast 65% der gesamten Population repräsentierten.

Zur Identifizierung interessanter Änderungen wurden in diesem Fall einfache Differenz-Parameter verwendet, die ausschließlich Änderungen der Regelbasis und des Supports der überwachten Muster betrachten. Wich die Anzahl permanenter Regeln in einer Periode um mehr als 20% vom Wert der Vorperiode ab oder änderte sich der Support eines Musters um mehr als 10 Prozentpunkte, wurde die Notwendigkeit einer neuen Mining-Session signalisiert. Gemäß dem ersten Kriterium wurden für das Muster M1 in den Perioden 5, 21 und 31 interessante Änderungen signalisiert und gemäß dem zweiten Kriterium in den Perioden 5, 6, 21, 22 und 52. Für das Muster M2 wurde dagegen ausschließlich in Periode 52 eine interessante Änderung angezeigt.

Tabelle 7.5 zeigt einen Ausschnitt der Änderungen nichtbeobachteter Muster in Periode 5. Um festzustellen, ob es sich wirklich um starke Änderungen handelt, wurden zusätzlich auch die Daten der jeweiligen Vorperiode analysiert. Während bei den ersten zwei Änderungen ausschließlich der Support des Musters betroffen ist, der in beiden Fällen um mehr als 50% abnimmt,

| Body | Head | Konfidenz | Support | Periode |
|---|---|---|---|---|
| 000000101 | 010000001 | 0,835 | 0,103 | 4 |
| 000000101 | 010000001 | 0,800 | 0,034 | 5 |
| 100100001 | 100010001 | 0,951 | 0,226 | 4 |
| 100100001 | 100010001 | 0,952 | 0,092 | 5 |
| 110000001 | 100010001 | 0,481 | 0,079 | 4 |
| 110000001 | 100010001 | 0,861 | 0,546 | 5 |

Tabelle 7.5: Auswahl der Musteränderungen in Periode 5

weist das letzte Muster sowohl für den Support als auch für die Konfidenz sehr große Zuwächse auf. Obwohl die extremen Änderungen des letzten Musters sicher eine Ausnahme darstellen, ergaben sich für die anderen überprüften Perioden vergleichbare Resultate.

Zusätzlich sollte die Fragestellung geklärt werden, ob aufgrund interessanter Änderungen in den Zeitreihen der beobachteten Muster auch auf interessante Änderungen der Regelbasis geschlossen werden kann, wie sie sich beispielsweise durch eine kontinuierliche Analyse mit Hilfe einer Mining-Software ergeben würden. Zu diesem Zweck wurden die Daten aller Perioden analysiert und die Anzahl der Muster in jeder Periode ermittelt. Abbildung 7.4 faßt die Ergebnisse dieses Schrittes der Analyse zusammen.

Es ist ersichtlich, daß sich für die Perioden, in denen durch die Überwachung der permanenten Regeln interessante Änderungen ermittelt werden konnten, auch deutliche Änderungen der Gesamtzahl der Regeln ergeben. In Periode 5 tritt sogar der Fall ein, daß die Anzahl der überwachten Regeln exakt der Gesamtzahl der in den Daten zu entdeckenden Regeln entspricht. Abbildung 7.4 zeigt jedoch auch, daß es eine ganze Reihe weiterer Perioden gibt, in denen die Regelbasis vergleichsweise starke Änderungen aufweist – insgesamt wurden zusätzlich 15 Perioden gefunden, in denen die Regelbasis Änderungen von mehr als 20% aufwies. Es ist außerdem auffällig, daß die Anzahl der beobachteten Muster im Vergleich zur Gesamtzahl der Muster in den jeweiligen Perioden spätestens ab Periode 31 sehr klein ist.

In der in [Baron u. a. 2003] beschriebenen Fallstudie wurde deshalb die Schlußfolgerung gezogen, daß die Überwachung nur der permanenten Muster unter Umständen nicht ausreicht, um wichtige Änderungen der Regelbasis zu erkennen. Deshalb wurde zusätzlich die im folgenden Abschnitt beschriebene

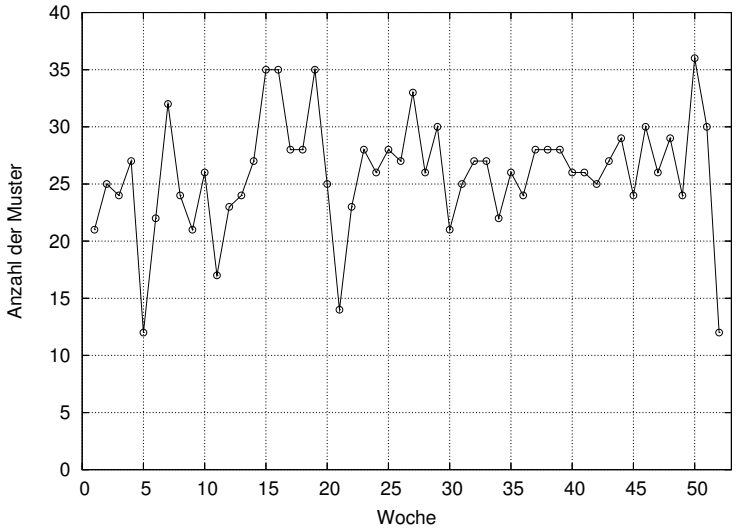


Abbildung 7.4: Entwicklung der Regelbasis

Analyse durchgeführt, bei der die Menge der zu überwachenden Muster auf geeignete Weise erweitert werden sollte.

### Überwachung häufiger Muster

Auch in diesem Fall wurde die Analyse in zwei Schritten durchgeführt, wobei nicht nur permanente Muster überwacht wurden, sondern alle Muster, die zur Gruppe der *häufigen* Muster gehören, d. h. deren Frequenz mindestens 0,75 betrug (vgl. Abschnitt 5.2.2). Hierbei kann in der ersten Periode ebenfalls nicht festgestellt werden, welche Muster letztlich in diese Kategorie fallen. Bei dem gewählten Verfahren zur Auswahl der zu beobachtenden Muster ist es darüber hinaus nicht sicher, ob alle zu überwachenden Muster bereits in den Ergebnissen der ersten Periode enthalten sind. Deswegen wurde die Mining-Phase auf die ersten zehn Perioden ausgedehnt. In der Monitoring-Phase wurden somit alle Muster überwacht, die in mindestens acht der zehn Trainingsperioden gefunden werden konnten.

Abbildung 7.5 zeigt die Änderungen der Anzahl überwachter Muster im Zeitverlauf. Die senkrechte Linie in der zehnten Woche bezeichnet das Ende der Trainingsphase. Insgesamt wurden in den ersten zehn Perioden 36 unterschiedliche Muster gefunden. Innerhalb der Trainingsphase wurden die

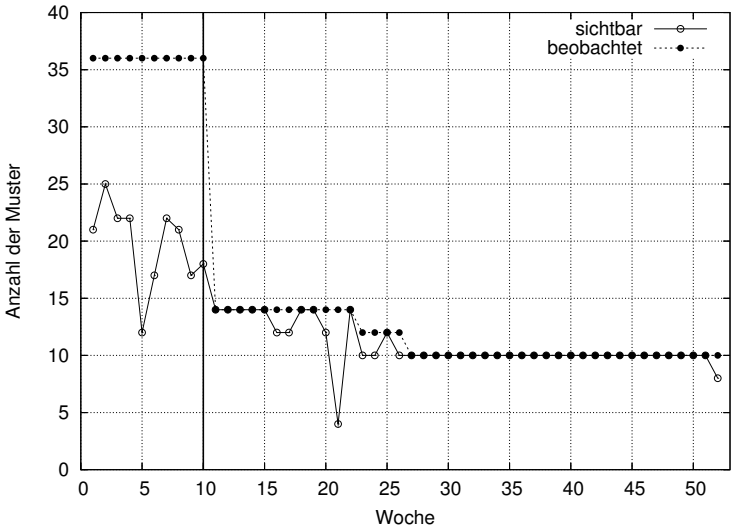


Abbildung 7.5: Entwicklung der Anzahl überwachter Muster

Zeitreihen der statistischen Eigenschaften *aller* Muster fortgeschrieben. Ab der elften Periode wurden dann nur jene Muster beobachtet, die bis zu diesem Zeitpunkt die erforderliche Frequenz von mindestens 0,75 aufwiesen. Zeigte ein Muster nach Ablauf des Trainings zu einem beliebigen Zeitpunkt nicht die erforderliche Frequenz, wurde es aus der Menge beobachteter Muster entfernt.[5]

Nach dem sehr starken Rückgang der Anzahl beobachteter Muster in Periode 11, der auf das Ende der Trainingsphase zurückzuführen ist, sind nur in den Perioden 23 und 27 weitere, sehr moderate Änderungen zu erkennen. Es wird deutlich, daß durch die geforderte Sichtbarkeit der Muster von 75% und die notwendige Trainingsphase deutlich weniger Änderungen der Anzahl beobachteter Muster zu verzeichnen sind. Die zweite Kurve in Abbildung 7.5 zeigt die Entwicklung der Anzahl beobachteter Muster, die auch *sichtbar* sind, d. h. die in der Mining-Anfrage festgelegten Anforderungen an Support,

[5]In diesem Fall wurde nicht der SAS Enterprise Miner verwendet, sondern der *Apriori*-Algorithmus aus dem Weka-Paket, der wesentlich leichter in eigene Anwendungen zu integrieren ist (vgl. Abschnitt 6.2.3). Bei diesem Algorithmus kann die untere Grenze für den Support jedoch nur bezogen auf die Gesamtzahl der Transaktionen angegeben werden. Um dennoch vergleichbare Resultate zu erhalten, wurde die Schwelle für den Support auf $\tau_s = 0,02$ festgelegt.

Konfidenz und Lift erfüllten. Aufgrund der weniger starken Einschränkung hinsichtlich der Frequenz entsprach dieser Wert nicht immer der Zahl der beobachteten Muster. Da es für den Anwender immer von besonderem Interesse ist, wenn ein neues Muster erscheint oder ein altes verschwindet, und die Änderungen der Anzahl beobachteter Muster allein wenig aussagekräftig sind, wurde diese zweite Kurve zusätzlich einbezogen, um auf interessante Änderungen der Regelbasis zu schließen. Von starken Änderungen in der Trainingsphase abgesehen konnten auf diese Weise insgesamt fünf Perioden identifiziert werden, in denen die Notwendigkeit einer neuen Mining-Session signalisiert wurde.

Abbildung 7.6 zeigt die Zeitreihen für den Support der beobachteten Muster. Auch in diesem Fall wurde bei Mustern, die den gleichen Zusammenhang beschreiben, nur das Muster mit der höheren Konfidenz überwacht. Die senkrechte Linie in Periode 10 bezeichnet wieder das Ende der Trainingsphase, die waagerechte Linie bei 0,02 die bei der Analyse verwendete untere Support-Schranke. Die Muster M6 bzw. M7 genügen der Mindestanforderung bezüglich der Frequenz nur bis Periode 22 bzw. 26, was zu den in Abbildung 7.5 dargestellten Änderungen der Anzahl überwachter Muster in den Perioden 23 bzw. 27 führt. Auch für diese Zeitreihen wurden alle Änderungen um mehr als 10 Prozentpunkte identifiziert, wobei jedoch nur zwei weitere Perioden gefunden wurden, in denen zusätzliche Mining-Sessions erforderlich wären.

Für die gesamte Regelbasis konnten dagegen insgesamt 18 Perioden ermittelt werden, in denen sich die Gesamtzahl der sichtbaren Muster im Vergleich zur jeweiligen Vorperiode um mindestens 20% änderte. Davon konnten jedoch nur zwei Perioden durch die Beobachtung der Zeitreihen und der Anzahl überwachter Muster gefunden werden. Wie sich auch schon im Fall der Überwachung permanenter Muster zeigte, ist offenbar kein direkter Zusammenhang zwischen den Zeitreihen der überwachten Muster und der Entwicklung der Regelbasis zu vermuten, und auch die Einbeziehung der Gruppe der häufigen Muster in die Überwachung scheint keine zusätzlichen Aufschlüsse diesbezüglich zu liefern. Um diese Vermutung zu überprüfen, wurden zwei weitere Analysen durchgeführt – eine, die den Zusammenhang zwischen der Anzahl beobachteter Muster und der Entwicklung der gesamten Regelbasis untersuchte, und eine weitere Analyse, die den Zusammenhang zwischen den Zeitreihen überwachter Muster und der Entwicklung der Regelbasis klären sollte.

Für die Zeitreihe der Musteranzahl in der Regelbasis wurde zunächst be-

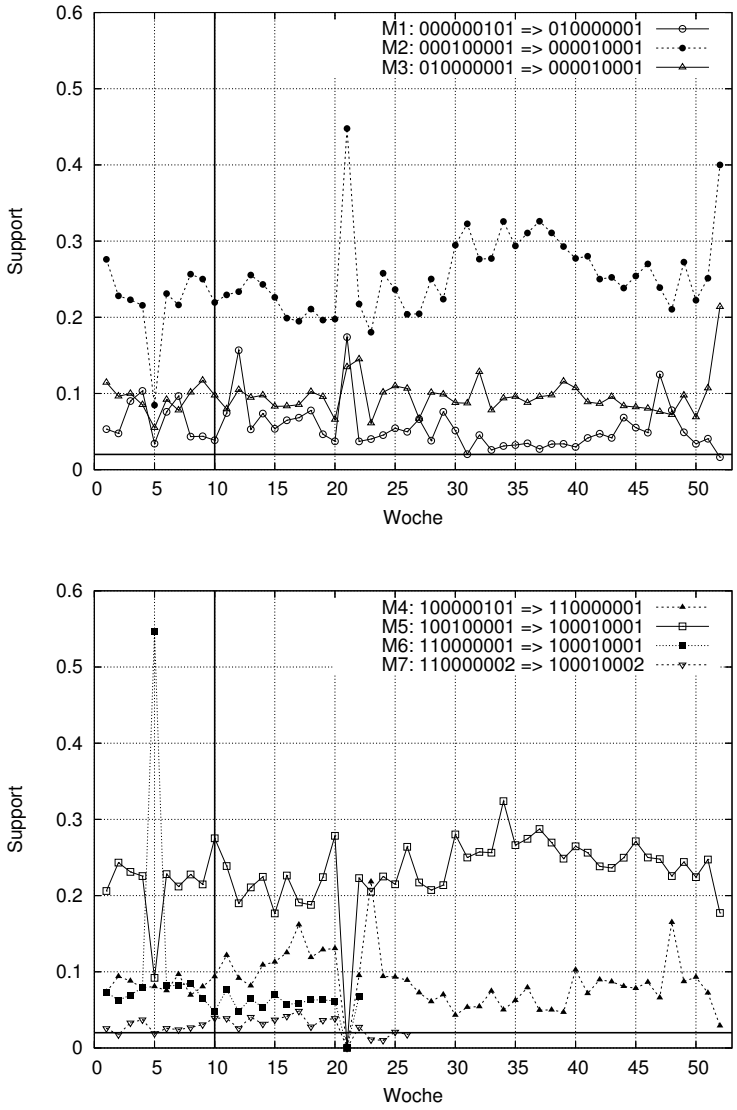


Abbildung 7.6: Zeitreihen des Supports überwachter Muster

stimmt, wie viele Muster in der jeweiligen Periode neu erschienen waren, wie viele aus der Vorperiode bekannt waren und wieviel Muster aus der Vorperiode nicht mehr gefunden werden konnten. Tabelle 7.6 zeigt die Ergebnisse dieser Auswertung für die Perioden 10 bis 15. So konnten z. B. in Periode 12

| | Anzahl der Muster | | | |
|---|---|---|---|---|
| Periode | gesamt | neu | alt | verschwunden |
| 10 | 18 | 4 | 14 | 3 |
| 11 | 16 | 2 | 14 | 4 |
| 12 | 18 | 4 | 14 | 2 |
| 13 | 19 | 3 | 16 | 2 |
| 14 | 18 | 2 | 16 | 3 |
| 15 | 35 | 17 | 18 | 0 |

Tabelle 7.6: Ausschnitt aus den Änderungen der Regelbasis

insgesamt 18 Muster gefunden werden, von denen 14 bereits aus der Vorperiode bekannt und vier neu waren. Zwei Muster aus der Vorperiode konnten nicht mehr gefunden werden.

Für die daraus resultierenden Zeitreihen und die Entwicklung der Anzahl beobachteter Regeln über alle Perioden wurden dann die Korrelationskoeffizienten bestimmt, die in Tabelle 7.7 dargestellt sind.

| | gesamt | neu | alt | verschwunden |
|---|---|---|---|---|
| sichtbar | −0,338 | −0,098 | −0,343 | −0,232 |
| beobachtet | −0,588 | −0,085 | −0,667 | −0,136 |

Tabelle 7.7: Korrelationskoeffizienten für die Zeitreihen der Regelbasis

Es ist ersichtlich, daß es zwischen der Entwicklung der Anzahl *sichtbarer* beobachteter Regeln und den Änderungen, denen die gesamte Regelbasis im Zeitverlauf ausgesetzt ist, keinen deutlichen Zusammenhang gibt. Ein geringfügig anderes Bild entsteht bei der Betrachtung der Entwicklung *aller* beobachteten Regeln. Hier zeigt sich zumindest eine leichte Korrelation mit der Gesamtzahl der beobachteten Muster, die für die Entwicklung der Anzahl der bereits aus der Vorperiode bekannten Muster noch etwas stärker ist. Interessanterweise haben alle Korrelationskoeffizienten ein negatives Vorzeichen. Die Entwicklung der Anzahl beobachteter Regeln scheint also tatsächlich kaum Anhaltspunkte für die Entwicklung der Regelbasis geben zu können.

In gleicher Weise wurden die Zeitreihen der statistischen Eigenschaften

und die Entwicklung der Regelbasis betrachtet. Für jede Eigenschaft wurde dazu die Korrelation mit den verschiedenen Aspekten der Entwicklung der Regelbasis bestimmt. Tabelle 7.8 faßt die Ergebnisse dieser Analyse zusammen.

| | | Anzahl der Muster | | | |
|---|---|---|---|---|---|
| | | gesamt | neu | alt | verschwunden |
| Eigenschaft | Support | 0/7 | 0/7 | 0/7 | 0/7 |
| | Konfidenz | 0/7 | 0/7 | 0/7 | 0/7 |
| | Lift | 0/7 | 0/7 | 0/7 | 1/6 |
| | abs. Support | 0/7 | 0/7 | 0/7 | 0/7 |

Tabelle 7.8: Korrelation der statistischen Eigenschaften beobachteter Muster und der Entwicklung der Regelbasis

Die erste Zahl gibt jeweils an, für wieviel Regeln der Betrag des Korrelationskoeffizienten bei der jeweiligen Kombination mindestens 0,5 betrug, die zweite Zahl, wie oft der Betrag kleiner als 0,5 war. In nur einem von 112 Fällen konnte überhaupt ein erkennbarer Zusammenhang ermittelt werden, der darüber hinaus mit $-0{,}576$ auch nur knapp über der absoluten Grenze von 0,5 lag. Auch bei der Betrachtung der Zeitreihen der statistischen Eigenschaften beobachteter Muster ist demnach kein direkter Zusammenhang mit der Entwicklung der Regelbasis zu erkennen. Genauer betrachtet, kann dieses Ergebnis jedoch nicht überraschen: Die Änderungen der häufigen Muster, die per Definition in mindestens 75% der Perioden auftreten, sollten keinen oder nur einen sehr schwachen Einfluß auf die Entwicklung der Regelbasis ausüben, da sie nur selten Auswirkungen auf die Sichtbarkeit des Musters haben. Andererseits sollten dann die Muster, deren Zeitreihen um die zur Musterentdeckung verwendeten Schwellwerte schwanken, die Regelbasis stärker beeinflussen.

Um auch diese Vermutung zu überprüfen, wurde die im Abschnitt 6.3.4 vorgestellte Methode zur vollständigen Überwachung der Muster auf die Menge der nichtbeobachteten Muster angewendet. Für alle Muster, die im Verlauf der Analyse in mindestens einer Periode gefunden werden konnten, wurden dazu die vollständigen Zeitreihen ihrer statistischen Eigenschaften bestimmt. Anschließend wurde für alle Zeitreihen die gleiche Korrelationsanalyse durchgeführt wie bereits für die beobachteten Muster. Die Ergebnisse dieser Un-

tersuchung sind in Tabelle 7.9 dargestellt.

| | | Anzahl der Muster | | | |
|---|---|---|---|---|---|
| | | gesamt | neu | alt | verschwunden |
| Eigenschaft | Support | 14/63 | 12/65 | 12/65 | 0/77 |
| | Konfidenz | 19/58 | 4/73 | 15/62 | 0/77 |
| | Lift | 24/53 | 0/77 | 2/75 | 0/77 |
| | abs. Support | 28/49 | 12/65 | 20/57 | 0/77 |

Tabelle 7.9: Korrelation der statistischen Eigenschaften der nichtbeobachteten Muster und der Entwicklung der Regelbasis

Insgesamt konnte für 162 von 1232 möglichen Kombinationen ein absoluter Korrelationskoeffizient von mindestens 0,5 gefunden werden. Das entspricht einem Anteil von rund 13% im Vergleich zu knapp einem Prozent im Fall der beobachteten Muster. Betrachtet man die Anzahl der Muster, für die der absolute Wert der Korrelation wenigstens einmal 0,5 betrug, fällt das Resultat noch eindeutiger aus: 56 von 77 Mustern bzw. fast 73% der Muster zeigen für mindestens eine Kombination einen deutlichen Zusammenhang, während es im Fall der beobachteten Muster nur eines von sieben ist (14,3%). Den deutlichsten Zusammenhang zwischen den statistischen Eigenschaften eines Musters und der Entwicklung der Regelbasis liefert demnach der absolute Support eines Musters (60), gefolgt von der Konfidenz und dem Support (jeweils 38). Da der absolute Support in der Mining-Anfrage nicht beschränkt wurde, hat die Wahl der unteren Grenze für Konfidenz und Support den größten Einfluß auf die Entwicklung der Regelbasis. Betrachtet man dagegen ausschließlich die Gesamtzahl vorhandener Regeln, hat die Auswahl der Schwelle für den Lift das größte Gewicht.

Zusätzlich sollte geklärt werden, ob ein Zusammenhang zwischen der Anzahl der Perioden, in denen ein Muster gefunden werden konnte, und der Stärke der Korrelation seiner Zeitreihen mit der Entwicklung der Regelbasis besteht. Dazu wurde für die Muster, die hohe Korrelationskoeffizienten aufwiesen, ermittelt, in wie vielen Perioden sie sichtbar waren. Tabelle 7.10 faßt die Ergebnisse dieser Prüfung zusammen.

Es zeigte sich, daß mehr als zwei Drittel der Muster, deren Zeitreihen hohe Korrelationskoeffizienten aufweisen, in weniger als der Hälfte der Perioden sichtbar sind. Die intuitive Vermutung, daß die Entwicklung der Regelbasis

| Perioden sichtbar | Anzahl der Muster |
|---|---|
| $50 < x \leq 52$ | 0 |
| $45 < x \leq 50$ | 4 |
| $40 < x \leq 45$ | 2 |
| $35 < x \leq 40$ | 2 |
| $30 < x \leq 35$ | 8 |
| $x \leq 30$ | 40 |

Tabelle 7.10: Anzahl der Perioden, in denen Muster mit starker Korrelation sichtbar sind

weitaus stärker von den temporären Mustern beeinflußt wird als von den häufigen Mustern, die in die Überwachung einbezogen wurden, scheint also offensichtlich zuzutreffen. Ein weiteres Indiz, das diese Schlußfolgerung untermauert, ist die Tatsache, daß das einzige beobachtete Muster, dessen Korrelationskoeffizient einen Betrag von mehr als 0,5 aufwies, in nur 39 Perioden sichtbar war und damit die festgelegte Grenze für die Frequenz von mindestens 0,75 gerade noch erreichte.

Der Überwachung nur einer Untermenge von Mustern lag die Annahme zugrunde, daß Änderungen nichtüberwachter Muster indirekt über Änderungen der überwachten Muster erkannt werden können. Um diese Annahme zu überprüfen, wurde – basierend auf den Ergebnissen der vollständigen Überwachung nichtbeobachteter Muster – eine Korrelationsanalyse der Zeitreihen der statistischen Eigenschaften der sieben beobachteten und der 77 nichtbeobachteten Muster durchgeführt, deren Ergebnisse in Tabelle 7.11 dargestellt sind.

| Eigenschaft | $\lvert \geq 0{,}5 \rvert$ | $\lvert < 0{,}5 \rvert$ |
|---|---|---|
| Support | 60 | 479 |
| Konfidenz | 35 | 504 |
| Lift | 46 | 493 |
| abs. Support | 61 | 478 |

Tabelle 7.11: Korrelation überwachter und nichtüberwachter Muster

Es ist zu erkennen, daß der Anteil der Zeitreihen, die einen absoluten

Korrelationskoeffizienten von mindestens 0,5 aufweisen, nur zwischen sechs und elf Prozent liegt. Interessant war jedoch, daß insgesamt für 67 der 77 nichtbeobachteten Muster jeweils für mindestens eine der statistischen Eigenschaften eine starke Verbindung mit den beobachteten Mustern gefunden werden konnte. Nur für zehn Muster bzw. knapp 13% der Muster in der Regelbasis konnte keine signifikante Korrelation gefunden werden. Von besonderem Interesse war nun natürlich die Frage, wie groß die *Bedeutung* dieser zehn Muster bezogen auf die Entwicklung der gesamten Regelbasis war. Es stellte sich heraus, daß alle zehn Muster ausschließlich in Periode 50 gefunden werden konnten, sie also gemäß den für Muster geltenden Stabilitätskriterien zur Gruppe der *temporären* Muster gehören (vgl. Abschnitt 5.2.2).

Unter bestimmten Voraussetzungen sind für den Anwender oft auch sogenannte Ausreißer von Interesse. Jedoch stellt sich bei dieser äußerst geringen Frequenz von knapp 0,02 die Frage, ob dies hier wirklich der Fall wäre: Interessiert sich der Nutzer tatsächlich für ein Muster, das über einen Zeitraum von einem Jahr nur einmal gefunden werden konnte? Wie so oft wird sich diese Frage endgültig nur unter Einbeziehung des Anwendungskontextes beantworten lassen. In diesem Fall handelte es sich um die in Tabelle 7.12 dargestellten Muster.[6]

Es wird deutlich, daß alle zehn Muster insgesamt nur drei unterschiedliche Items betreffen (`100000102`, `100010002` und `101000002`). Alle diese Muster beschreiben Nachrichten, die ein Absender innerhalb der Fakultät während der Arbeitszeit an einen Institutsmitarbeiter und einen externen Empfänger geschickt hat. Die beiden letzten Muster scheinen eigentlich den gleichen Zusammenhang widerzuspiegeln, ihr Support ist jedoch geringfügig höher als bei den anderen Mustern. Der höhere Support folgt aus der allgemeineren Aussage des Musters, daß E-Mails, die aus der Fakultät an zwei Empfänger geschickt werden, mindestens auch einen Empfänger im Institut haben. Dieses Muster gilt genauso für den Fall, daß die Nachricht an zwei Mitarbeiter des Instituts geschickt wurde. Da das erste der beiden Muster eine Konfidenz von 100% aufweist, ist es offenkundig, daß es keine Nachrichten an zwei Empfänger gibt, von denen nicht mindestens einer am Institut arbeitet. Vom Anwendungskontext her sind diese Muster eigentlich sehr interessant, da der

[6] Die erste Spalte (`URID`) beinhaltet die eindeutige Kennung der Regel (*Unique Rule Identifier*), die gemäß der im Abschnitt 4.2.1 beschriebenen Bildungsvorschrift aus der linken und rechten Seite des Musters bestimmt wird. Der doppelte Unterstrich wird dabei verwendet, um Body und Head des Musters zu trennen; der einfache Unterstrich trennt die Elemente innerhalb der linken bzw. rechten Seite.

| URID | Konfidenz | Support | Lift | Anzahl |
|---|---|---|---|---|
| 101000002_100000102__100010002 | 1,000 | 0,0882 | 6,52 | 239 |
| 101000002_100010002__100000102 | 0,988 | 0,0882 | 6,51 | 239 |
| 100010002_100000102__101000002 | 0,605 | 0,0882 | 6,78 | 239 |
| 101000002__100010002_100000102 | 0,988 | 0,0882 | 6,78 | 239 |
| 100010002__101000002_100000102 | 0,575 | 0,0882 | 6,52 | 239 |
| 100000102__101000002_100010002 | 0,582 | 0,0882 | 6,51 | 239 |
| 101000002__100000102 | 0,988 | 0,0882 | 6,51 | 239 |
| 100000102__101000002 | 0,582 | 0,0882 | 6,51 | 239 |
| 101000002__100010002 | 1,000 | 0,0893 | 6,52 | 242 |
| 100010002__101000002 | 0,582 | 0,0893 | 6,52 | 242 |

Tabelle 7.12: Muster ohne signifikante Korrelation

Administrator des Mail-Servers aufgrund der Ausprägung ihrer statistischen Eigenschaften davon ausgehen kann, daß sein Server zumindest nicht von Mitgliedern der Fakultät unerlaubt benutzt wird. In allen anderen Perioden wiesen diese Muster jedoch nicht den notwendigen Support auf, um in den Ergebnissen des Miners zu erscheinen.

### Effizienzbetrachtungen

Ein weiterer Aspekt dieser Fallstudie bestand darin, zu klären, ob die SQL-basierte Fortschreibung der gefundenen Muster effizienter ist als ihre Aktualisierung mit Hilfe einer Mining-Software. Bezüglich der Genauigkeit der Ergebnisse sind für die vorgeschlagene Methode sicher keine Einschränkungen zu erwarten. Werden die gleichen Parameter wie bei einer konventionellen Analyse verwendet, ist auch mit identischen Resultaten zu rechnen.

Für den erforderlichen Rechenaufwand sollte sich jedoch aufgrund der verschiedenen Zielstellungen ein grundlegend anderes Bild ergeben. Eine Mining-Software sucht – abgesehen von Werkzeugen, die eine dedizierte Anfragesprache anbieten – grundsätzlich nach allen Mustern, die den in der Mining-Anfrage spezifizierten Kriterien genügen. Wichtigste Charakteristik einer solchen Suche ist, daß sie zunächst völlig ungerichtet erfolgt. Ein Algorithmus zur Warenkorbanalyse wie z. B. *Apriori* sucht in der ersten Iteration nach allen Items, die der Mindestanforderung an den Support genügen (vgl. Abschnitt 2.2). Da zu diesem Zeitpunkt unbekannt ist, welche Items diesem

Kriterium genügen, muß der Algorithmus dafür den Support aller Items bestimmen. Bei der SQL-basierten Ermittlung der statistischen Eigenschaften dagegen entfällt dieser Schritt. Anfangs ist auch in diesem Fall unbekannt, welche Muster den in der Mining-Anfrage formulierten Kriterien genügen. Da die Statistiken jedoch nur für eine ausgewählte Menge von Mustern bestimmt werden, erfolgt ihre Berechnung grundsätzlich gerichtet.

Zur Beurteilung der Effizienz der SQL-basierten Fortschreibung der Muster wurde ein Datensatz verwendet, der etwa 6000 Transaktionen enthielt. In diesem Datensatz wurden 20 Muster gefunden, für die die Werte der statistischen Eigenschaften aktualisiert werden sollten. Während die Performanz konventioneller Algorithmen im wesentlichen von der Anzahl der Transaktionen abhängt, ist für die vorgeschlagene Methode eher die Azahl der zu aktualisierenden Muster von Interesse. Ohne Zweifel muß auch die Datenbank die vollständige Transaktionstabelle auslesen, um die Anzahl der Transaktionen zu ermitteln, die ein bestimmtes Item enthalten. Zum einen ist ein relationales Datenbanksystem jedoch exakt für diese Operationen optimiert, zum anderen muß ein Mining-Algorithmus wie *Apriori* die Tabelle mehrmals einlesen, insbesondere wenn sie sehr viele Transaktionen enthält und der verfügbare Arbeitsspeicher ein limitierender Faktor ist. Zur Aktualisierung eines Musters müssen insgesamt drei SQL-Anfragen gestellt werden: eine, die die Gesamtzahl der Transaktionen in der jeweiligen Periode bestimmt; eine, die die Anzahl der Transaktionen bestimmt, die die linke Seite der Assoziationsregel enthalten; eine Anfrage, die die Anzahl der Transaktionen bestimmt, die alle Items der Regel enthalten (vgl. Beispiel 4.4). Aus den Ergebnissen dieser drei Anfragen können dann alle statistischen Maßzahlen berechnet werden, die für ein Muster erfaßt werden. Da die Gesamtzahl der Transaktionen in einer Periode nur einmal für alle zu aktualisierenden Muster ermittelt werden muß, hängt die Anzahl der notwendigen Anfragen linear von der Anzahl der zu aktualisierenden Muster ab. Sei $n$ die Anzahl der Muster, für die die statistischen Eigenschaften bestimmt werden sollen, dann ist die Komplexität der SQL-basierten Aktualisierung der Muster gerade $O(2n+1)$.

Für die Berechnung der statistischen Eigenschaften der 20 Muster mußten insgesamt $2 \times 20 + 1$ Anfragen an die Datenbank gestellt werden, für die jeweils Abarbeitungszeiten zwischen 0ms und 5ms anfielen. Im Durchschnitt wurde für eine Anfrage 1ms und für die Abarbeitung aller 41 Anfragen 0,105s benötigt. Im Vergleich dazu benötigte der Miner durchschnittlich 2,0045s, um den gesamten Datensatz zu analysieren. Die SQL-basierte Aktualisierung der Muster benötigte somit nur etwa den zwanzigsten Teil der Zeit, die der Miner

für die Analyse brauchte. Obwohl in diese Betrachtung nicht die Zeit einbezogen ist, die für die Generierung der SQL-Anfragen oder die Nachbearbeitung der vom Miner gefundenen Muster benötigt würde, verschaffen diese Zahlen einen Eindruck von den erreichbaren Einsparungen an benötigter Rechenzeit.

### 7.1.3 Fazit

In dieser Fallstudie sollte überprüft werden, ob die Überwachung einer ausgewählten Teilmenge von Mustern ausreicht, um wichtige Änderungen der Regelbasis zu erkennen. Neben der Klärung der Frage, in welchem Ausmaß Änderungen der beobachteten Muster Einfluß auf die Entwicklung der Regelbasis haben, sollte hauptsächlich ermittelt werden, ob ausgehend von den starken Änderungen der überwachten Muster auf starke Änderungen in der Menge der nichtbeobachteten Muster geschlossen werden kann. Zusätzlich war in diesem Zusammenhang natürlich die Quantifizierung des erreichbaren Vorteils von Interesse.

Das entstandene Bild wies eine deutliche Zweiteilung auf. Zum einen wurde festgestellt, daß die Überwachung einer Teilmenge der Muster kaum Rückschlüsse auf die Dynamik der Regelbasis erlaubt. Die Zeitreihen der verschiedenen Aspekte der Entwicklung der Regelbasis wiesen kaum Korrelationen mit den Zeitreihen der statistischen Eigenschaften überwachter Muster auf. Wie die weitere Untersuchung zeigte, war dieses Ergebnis jedoch nicht überraschend und durch die Art der Auswahl der zu beobachtenden Muster bedingt: Da ausschließlich Muster überwacht wurden, die in allen bzw. sehr vielen Perioden vertreten waren, konnten die Zeitreihen ihrer statistischen Eigenschaften nur einen geringen Einfluß auf die Entwicklung der Regelbasis haben. Häufige Muster können naturgemäß nur sehr wenige Änderungen aufweisen, die ihre Sichtbarkeit beeinflussen, da sie andernfalls kaum häufig sein könnten. Gerade diese Art von Änderungen beeinflußt jedoch die Dynamik der Regelbasis.

Auf der anderen Seite zeigten die Zeitreihen der statistischen Eigenschaften überwachter und nichtüberwachter Muster relativ starke Zusammenhänge. Nur für einen sehr kleinen Teil der nichtüberwachten Muster konnten keine deutlichen Beziehungen ermittelt werden. Insofern bestätigte sich die der Analyse zugrunde liegende Hypothese, daß Änderungen überwachter Muster Rückschlüsse auf Änderungen der nichtbeobachteten Muster zulassen. Darüber hinaus zeigte sich, daß die SQL-basierte Aktualisierung der Muster-Statistiken deutlich effizienter durchgeführt werden kann, als es

mit einer Mining-Software möglich wäre. Letztlich kann jedoch auch dieses Ergebnis nicht überraschen: Relationale Datenbanken verfügen über eine langjährige Entwicklungsgeschichte und sind unter anderem genau für diese Art von Zugriffen optimiert.

Da die vorgeschlagene Aktualisierung der Muster keine direkten Rückschlüsse auf die Entwicklung der Regelbasis zuläßt, ist die Entdeckung neuer Muster immer nur in jenen Perioden möglich, in denen aufgrund starker Änderungen der beobachteten Muster eine neue Mining-Session durchgeführt wird. Ist der Anwender daran interessiert, zu jedem Zeitpunkt eine *vollständige* Übersicht über alle in den Daten vorhandenen Muster zu gewinnen, eignet sich die vorgeschlagene Methode also nur eingeschränkt, da Rückschlüsse auf die Entwicklung der Regelbasis nur indirekt möglich und mit einer gewissen Unsicherheit verbunden wären. Allerdings erhebt sich bei dieser Zielsetzung die Frage, ob die reguläre Nutzung einer Mining-Software für den Anwender nicht ohnehin vorteilhafter wäre. Ist es dagegen wichtig, grundlegende Änderungen der statistischen Eigenschaften erkennen zu können, bietet die vorgeschlagene Methode einen effizienten Weg, dieses Ziel zu erreichen. Ohne Zweifel ist die SQL-basierte Fortschreibung der Muster-Statistiken jedoch immer dann besonders gut geeignet, wenn bereits *vorher* bekannt ist, welche Muster überwacht werden sollen, z. B. weil sie durch den Anwendungskontext vorgegeben sind. Der Nutzer braucht jetzt nur noch den Inhalt der interessierenden Muster anzugeben – und ihre Statistiken werden dann direkt aus den zugrunde liegenden Transaktionsdaten bestimmt. Da bei dieser Verfahrensweise die aufwendige, ungerichtete Suche eines Mining-Algorithmus überflüssig wird, ergeben sich für die Abarbeitungszeit deutliche Einsparungspotentiale.

## 7.2 Identifizierung interessanter Änderungen in Navigationsmustern

Im Rahmen dieser Fallstudie sollten die im Kapitel 5 vorgestellten Methoden zur Erkennung interessanter Musteränderungen Anwendung finden, wobei eine Unterscheidung in kurz- und langfristige Musteränderungen vorgenommen werden sollte [Baron und Spiliopoulou 2003]. Zur Vereinfachung galt dabei die Voraussetzung, daß eine langfristige Änderung immer dann vorliegt, wenn der Wert der betrachteten Eigenschaft nicht unmittelbar in der jeweils folgenden Periode zu seinem ursprünglichen Niveau zurückkehrte. Außerdem

sollten für alle Muster, die eine interessante Änderung zeigten, jene Bestandteile ermittelt werden, die ebenfalls interessante Änderungen aufwiesen und somit potentiell zur beobachteten Musteränderung beigetragen hatten.

### 7.2.1 Untersuchter Datensatz

Bei den untersuchten Daten handelt es sich um die Protokolldatei eines WWW-Servers, in der die Zugriffe auf ein nichtkommerzielles Informationsangebot zum Thema Patente über einen Zeitraum von acht Monaten erfaßt waren. Im allgemeinen zeichnet ein WWW-Server eine ganze Reihe von Attributen auf, die einen einzelnen Zugriff näher beschreiben, z. B. die IP-Adresse des zugreifenden Rechners, Datum und Uhrzeit des Zugriffs, die genaue Adresse des aufgerufenen Objektes, den Status des Zugriffs, die übertragene Datenmenge und vieles mehr. Darüber hinaus werden nicht nur die Zugriffe auf die einzelnen Seiten des Servers aufgezeichnet, sondern auch Aufrufe von Bildern, Navigationshilfen und Menüs etc. In diesem Fall lag das Protokoll jedoch nicht in seiner Rohform vor, sondern hatte im Rahmen einer früheren Analyse schon umfangreiche Maßnahmen zur Datenvorbereitung durchlaufen. So waren bereits alle für die Ermittlung von Navigationsmustern unnötigen und störenden Zugriffe auf die erwähnten Bilder, Navigationshilfen und Menüs entfernt und – eine wesentliche Voraussetzung für die vorgesehene Analyse – die verbliebenen Seitenzugriffe sogenannten Sitzungen, zusammengehörige Seitenaufrufe, die jeweils einem Besuch eines Nutzers entsprechen, zugeordnet. Zusätzlich wurden alle Seiten, die über den Server zugänglich waren, einer inhaltsbasierten Konzepthierarchie zugeordnet, deren Aufbau aus Abbildung 7.7 hervorgeht. Somit wurde nicht mehr die Navigation der Nutzer zwischen den verschiedenen Seiten des Servers untersucht, sondern zwischen den verschiedenen Informationsangeboten, denen diese Seiten zugeordnet waren. Zugriffe auf die Seiten, die dem Konzept `I2` zugeordnet waren, wurden nicht in die hier beschriebene Analyse einbezogen, da sie über einen anderen WWW-Server erreichbar waren.

### 7.2.2 Datenanalyse

Wie bereits erwähnt, wird jeder einzelne Seitenabruf durch mehrere Attribute beschrieben. Für die durchzuführende Analyse waren jedoch nur drei Felder von Interesse, die Adresse der aufgerufenen Seite bzw. das Konzept, zu dem die Seite gehörte, das Datum, an dem der Zugriff erfolgte, und die im Rahmen

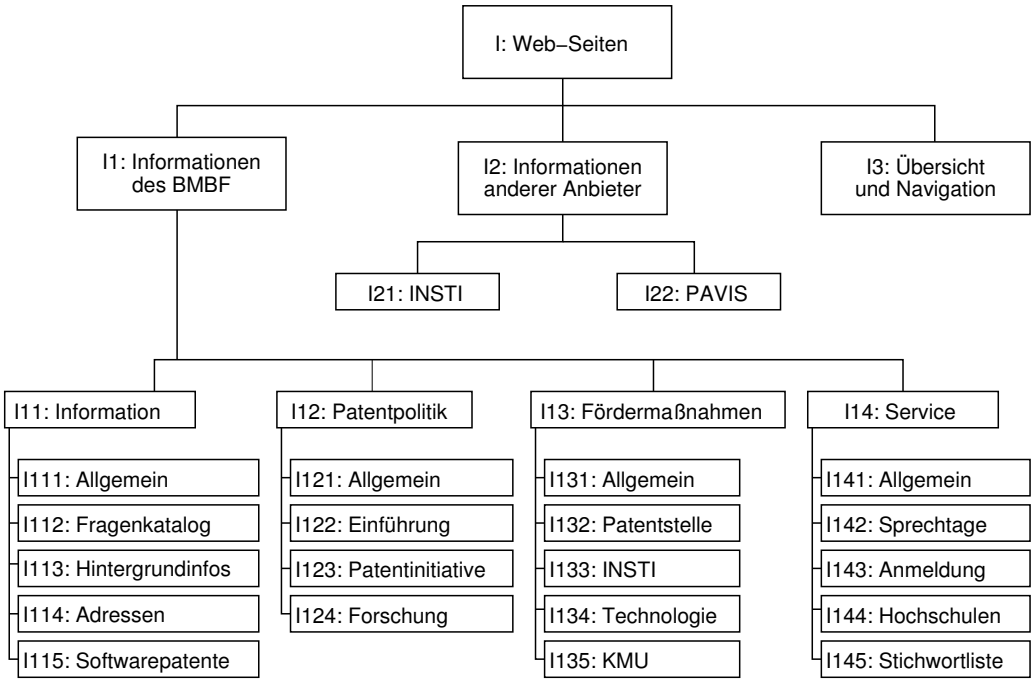


Abbildung 7.7: Inhaltsbasierte Taxonomie für die Seiten des WWW-Servers

der Datenvorbereitung erzeugte Kennung der Sitzung, innerhalb derer auf das jeweilige Konzept zugegriffen wurde. Insgesamt enthielt das Protokoll rund 62 500 Zugriffe, die von etwa 19 000 Sitzungen stammten, und wurde auf monatlicher Basis geteilt, um für die resultierenden Perioden mit Hilfe einer Warenkorbanalyse zu bestimmen, auf welche Konzepte innerhalb der gleichen Sitzung häufig gemeinsam zugegriffen wurde. Für die Analyse wurden ein minimaler Support $\tau_s = 2{,}5\%$ und eine minimale Konfidenz von $\tau_c = 80\%$ verwendet.

### Überblick

Tabelle 7.13 gibt einen allgemeinen Überblick über die Anzahl der Zugriffe, der Sitzungen, der häufigen Konzepte und über die Anzahl der gefundenen Muster in den jeweiligen Perioden.

Die erste Periode der Analyse entspricht dem Monat Oktober. Obwohl die Anzahl der Zugriffe und Sitzungen in den Perioden 3 und 4 (Dezember und

| Periode | Zugriffe | Sitzungen | Konzepte | Muster |
|---|---|---|---|---|
| 1 | 8335 | 2547 | 20 | 22 |
| 2 | 9012 | 2600 | 20 | 39 |
| 3 | 6008 | 1799 | 20 | 26 |
| 4 | 4188 | 1222 | 21 | 24 |
| 5 | 9488 | 2914 | 20 | 14 |
| 6 | 8927 | 2736 | 20 | 15 |
| 7 | 7401 | 2282 | 20 | 13 |
| 8 | 9210 | 3014 | 20 | 11 |

Tabelle 7.13: Allgemeiner Überblick über das Zugriffsprotokoll

Januar) stark abnimmt, bleibt die Anzahl der häufigen Konzepte und Muster in diesen beiden Monaten – abgesehen von der Rückkehr zum ursprünglichen Niveau nach der starken Zunahme in der zweiten Periode – ungefähr auf gleicher Höhe. In der zweiten Hälfte des Analysezeitraums nimmt dagegen die Anzahl der Regeln merklich ab, obwohl die Anzahl der Zugriffe und Sitzungen auf relativ hohem Niveau bleibt. Eine solche Entwicklung könnte z. B. auf einen Lernprozeß bei den Nutzern der Website hindeuten: Nachdem die Inhalte des Servers erkundet wurden, folgt eine Phase, in der gezielt ausgewählte Informationen abgerufen werden. Für diese Hypothese spricht, daß es in der zweiten Analysehälfte nur vier Muster gibt, die zum ersten Mal gefunden wurden, während in der ersten Hälfte bereits 54 der insgesamt 58 unterschiedlichen Muster gefunden werden konnten. Zudem beträgt die Anzahl unterschiedlicher Regeln in der zweiten Hälfte nur 25 und liegt damit nicht einmal bei 50% der Anzahl in den ersten vier Perioden. Insgesamt gab es 33 Muster, die in der zweiten Hälfte des Analysezeitraums nicht mehr erschienen. Auch wenn a posteriori nicht mehr festgestellt werden kann, ob es sich tatsächlich um einen Lernprozeß handelte, ist in der zweiten Hälfte der Analyse eine deutliche Fokussierung des Nutzerverhaltens zu erkennen.

Tabelle 7.14 zeigt die Entwicklung der Regelbasis über den Analysezeitraum hinweg. Neben der Gesamtzahl der Muster in den jeweiligen Perioden ist angegeben, wieviel Muster im Hinblick auf den gesamten Analysezeitraum zum ersten Mal gefunden werden konnten (Spalte *unbekannt*), wie viele Muster bezogen auf die jeweilige Vorperiode neu (Spalte *bekannt*) bzw. bekannt (Spalte *alt*) waren und wie viele Muster aus der Vorperiode nicht mehr ge-

funden werden konnten (Spalte *verschwunden*).

| | Anzahl der Muster | | | | |
|---|---|---|---|---|---|
| Periode | *gesamt* | *unbekannt* | *bekannt* | *alt* | *verschwunden* |
| 1 | 22 | 22 | – | – | – |
| 2 | 39 | 27 | – | 12 | 10 |
| 3 | 26 | 4 | 9 | 13 | 26 |
| 4 | 24 | 1 | 11 | 12 | 14 |
| 5 | 14 | – | 1 | 13 | 11 |
| 6 | 15 | 1 | 5 | 9 | 5 |
| 7 | 13 | 2 | 3 | 8 | 7 |
| 8 | 11 | 1 | 2 | 8 | 5 |

Tabelle 7.14: Entwicklung der Regelbasis

Auch aus dieser Übersicht geht hervor, daß die Dynamik der Regelbasis in der zweiten Hälfte der Analyse deutlich abnimmt. Die Anzahl der Muster, die neu hinzukommen bzw. nicht mehr gefunden werden können, nimmt stark ab. Dies gilt insbesondere für die Gesamtzahl neuer Muster in einer Periode, d. h. der Summe der unbekannten Muster und der Anzahl jener Muster, die im Vergleich zur jeweiligen Vorperiode hinzugekommen sind. Das ist besonders interessant, weil dieser Zeitraum eigentlich eine höhere Anzahl an Sitzungen umfaßt. Es ist außerdem ersichtlich, daß bereits in der zweiten Periode 49 der insgesamt 58 unterschiedlichen Muster bekannt sind.

Um die Stabilität der gefundenen Muster einschätzen zu können, wurden die Muster im Hinblick auf den relativen Anteil der Perioden, in denen sie sichtbar waren, gruppiert. Es zeigte sich, daß nur 11 der 58 Regeln häufig auftraten, d. h. in mindestens sechs Perioden vertreten waren (vgl. Tabelle 7.15). Wegen der großen Anzahl der Muster, die nur in ein oder zwei Perioden sichtbar waren, gab es selbst in aufeinanderfolgenden Perioden starke Änderungen in der Regelbasis, insbesondere in der ersten Hälfte des Analysezeitraums (vgl. Tabelle 7.14).

### Identifizierung interessanter Änderungen

Wie im Abschnitt 5.3 ausgeführt wurde, ist die statische Ermittlung des Interesses, wie sie z. B. bei der Bestimmung interessanter Muster Verwendung

| Perioden | Muster | Anteil | Gruppe |
|---|---|---|---|
| 8 | 3 | 100.0 | permanente Muster |
| 7 | 4 | 87.5 | häufige Muster |
| 6 | 4 | 75.0 | |
| 5 | 2 | 62.5 | temporäre Muster |
| 4 | 2 | 50.0 | |
| 3 | 6 | 37.5 | |
| 2 | 15 | 25.0 | |
| 1 | 22 | 12.5 | |

Tabelle 7.15: Stabilität der gefundenen Muster

findet, für die Überwachung von Mustern nicht ausreichend. So ist es für den Anwender im allgemeinen von besonderem Interesse, welche Muster in einer Periode zusätzlich zu den schon bekannten gefunden wurden und welche Muster nicht mehr gefunden werden konnten. Auch die im Abschnitt 5.3.4 vorgestellten Heuristiken betrachten die Dynamik der Regelbasis, wobei zusätzliche Kriterien einbezogen werden, die eine Bewertung der *Stärke* der gefundenen Änderungen erlauben. Bei der Verwendung der verschiedenen Heuristiken muß natürlich beachtet werden, daß sie all jene Änderungen für interessant befinden werden, welche die für die jeweilige Heuristik definierten Kriterien erfüllen. Für die Klärung der Frage, ob eine für interessant befundene Änderung tatsächlich auch das Interesse des Anwenders findet, ist folglich eine Interpretation des Inhalts der Muster, für die interessante Änderungen gefunden wurden, notwendig.

Tabelle 7.16 stellt die Ergebnisse für die Anwendung der Signifikanztests und der verschiedenen Heuristiken gegenüber. Ein direkter Vergleich der Werte ist dabei nur bedingt aussagefähig, da für die Ermittlung interessanter Änderungen unterschiedliche Kriterien Anwendung finden. Die Gegenüberstellung ist jedoch geeignet, eine Übersicht zu vermitteln, welche Änderungen von den jeweiligen Methoden identifiziert werden. Die zweite Spalte der Tabelle enthält die Anzahl der Änderungen, die der Erkennung interessanter Änderungen jeweils zugrunde liegen, und die dritte Spalte die Anzahl der Änderungen, die gemäß der jeweiligen Methode interessant waren. In der vierten Spalte ist die Anzahl der interessanten Änderungen aufgeführt, die auch signifikant waren, und in der letzten Spalte die Anzahl der langfristigen

| Heuristik | betrachtet | interessant | signifikant | langfr. |
|---|---|---|---|---|
| Signifikanztest (sichtbar) | 75 | 10 | 10 | 3 |
| Signifikanztest (beobachtet) | 142 | 17 | 17 | 11 |
| Signifikanztest (alle) | 344 | 48 | 48 | 32 |
| stabilitätsbasiert | 178 | 33 | 10 | 6 |
| intervallbasiert | 344 | 16 | 12 | 4 |
| korridorbasiert | 178 | 79 | 22 | 10 |

Tabelle 7.16: Gegenüberstellung der Resultate für die verschiedenen Heuristiken

signifikanten Änderungen.[7]

Die erste Zeile zeigt die Ergebnisse der Signifikanztests, die für alle *sichtbaren* Muster durchgeführt wurden. Insgesamt gab es über den gesamten Analysezeitraum 164 *Fälle*, wobei ein Fall dem Auftreten eines Musters in einer Periode entspricht. Insgesamt können von den 164 Fällen aber höchstens 142 untersucht werden, da in der ersten Periode die für die Teststatistik benötigten Werte der Vorperiode nicht vorliegen. In diesem Fall wurden jedoch ausschließlich jene Änderungen auf ihre Signifikanz überprüft, die mit Hilfe konventioneller Mining-Software ermittelt werden konnten. Das bedeutet im einzelnen, daß eine Änderung nur dann auf ihre Signifikanz untersucht wurde, wenn das Muster auch in der Vorperiode sichtbar war, da sonst nicht alle Werte für die Teststatistik vorliegen, und daß eine langfristige Änderung nur dann gefunden werden konnte, wenn das betreffende Muster (in diesem Fall) in mindestens drei aufeinanderfolgenden Perioden in den Ergebnissen zu finden war. Unter diesen Voraussetzungen wurden insgesamt 75 Fälle untersucht, wobei 10 signifikante Änderungen gefunden werden konnten, 3 davon langfristiger Natur.

Die zweite Zeile zeigt die Ergebnisse der Signifikanztests, die für alle *beobachteten* Musteränderungen durchgeführt wurden. In diesem Fall wurde die Einschränkung bezüglich Verfügbarkeit der statistischen Eigenschaften teilweise aufgehoben: War der Support der Vorperiode nicht bekannt, weil das Muster nicht gefunden werden konnte, wurde er aus den zugrunde liegenden Transaktionsdaten bestimmt. In gleicher Weise wurde auch bei der

[7] Den zugrunde liegenden Kriterien entsprechend stimmen bei den Signifikanztests die Werte in den Spalten 3 und 4 überein.

Bestimmung langfristiger Änderungen vorgegangen. Insgesamt konnten in diesem Fall 17 signifikante Support-Änderungen gefunden werden, von denen immerhin schon 11 Änderungen langfristiger Natur waren. Die dritte Zeile von Tabelle 7.16 enthält die Anzahl signifikanter Änderungen, wenn *alle* Musteränderungen berücksichtigt werden. Dieser Vorgehensweise folgend, beginnt die Beobachtung eines Musters wie bei den Heuristiken mit seinem ersten Erscheinen. Abgesehen von den Änderungen, die *vor* der ersten Sichtbarkeit eines Musters zu verzeichnen wären, können auf diese Weise alle signifikanten Änderungen ermittelt werden. Im einzelnen konnten in diesem Fall 48 signifikante Änderungen gefunden werden, von denen 32 langfristig waren. Spätestens jetzt wird deutlich, daß durch die Verwendung eines Monitors, der auch die Basisdaten in die Analyse einbezieht, zusätzliche Erkenntnisse gewonnen werden können. So wurden in diesem Fall auch jene Änderungen verfolgt, die dazu geführt haben, daß alte Muster nicht mehr gefunden werden konnten bzw. neue Muster hinzukamen.

Die Heuristiken betrachten, wie bereits erwähnt, alle Änderungen eines Musters ab seinem ersten Erscheinen. Während die intervallbasierte Heuristik jedoch von der ersten Periode an interessante Musteränderungen identifizieren kann, benötigen die stabilitätsbasierte Gruppierung und die korridorbasierte Heuristik eine gewisse Trainingsphase, da die für die Erkennung interessanter Änderungen verwendeten Kriterien anfangs sehr stark auf Schwankungen reagieren (vgl. Abschnitt 5.3.4). Aufgrund der begrenzten Zahl an Perioden wurden deshalb die jeweils ersten vier Perioden als Lernphase verwendet und erst in der fünften Periode begonnen, interessante Musteränderungen zu identifizieren.[8] Aus diesem Grund ist für diese beiden Heuristiken eigentlich mit einer geringeren Anzahl interessanter Änderungen zu rechnen. Zu Vergleichszwecken wurden die interessanten Änderungen, die mit Hilfe der Heuristiken bestimmt wurden, zusätzlich auch auf ihre Signifikanz überprüft.

Mit Hilfe der stabilitätsbasierten Gruppierung wurden insgesamt 33 interessante Änderungen gefunden, von denen 10 signifikant waren, 6 davon langfristiger Natur. Abbildung 7.8 zeigt beispielhaft die Zeitreihe der Frequenz eines Musters. Die waagerechten Linien bei 0,5, 0,75 und 0,9 markieren die Intervallgrenzen der Frequenz, die senkrechte Linie in Periode 4 das Ende der Trainingsphase. Das Muster wurde in der ersten Periode gefunden. In

[8] Die Lernphase wird dabei für jedes Muster individuell bestimmt, d. h., für ein Muster, das in Periode 2 zum ersten Mal auftritt, werden z. B. erst ab Periode 6 interessante Änderungen identifiziert.

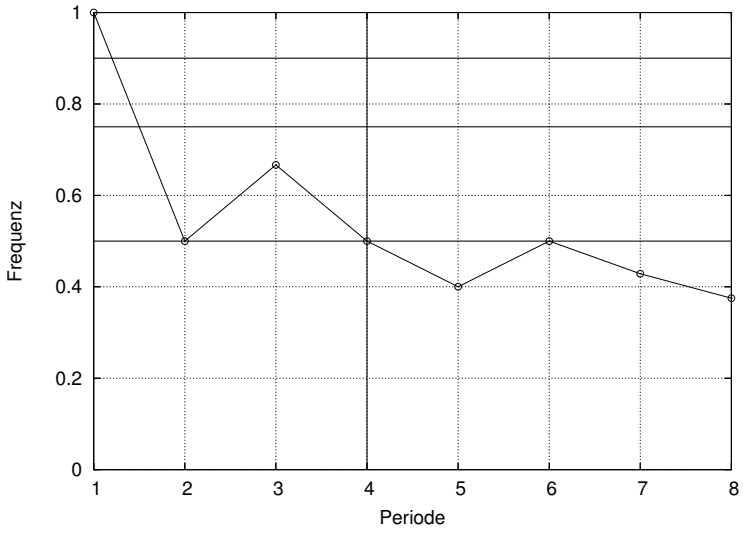


Abbildung 7.8: Zeitreihe der Frequenz eines Musters

der zweiten Periode ist es nicht sichtbar, weswegen die Frequenz auf 0,5 fällt. In der nächsten Periode ist das Muster wieder anwesend, und die Frequenz steigt wieder an. In den nächsten zwei Perioden kann das Muster erneut nicht gefunden werden. Mit dem Ende der Lernphase in Periode 4 weist das Muster eine Frequenz von 0,5 auf. Da es in Periode 5 ebenfalls nicht gefunden wurde, fällt die Frequenz weiter, und es wird ein erster Gruppenwechsel signalisiert. In der nächsten Periode ist das Muster wieder in den Daten vertreten und wechselt erneut die Gruppe, da die Frequenz wiederum auf 0,5 steigt. In den verbleibenden Perioden des Analysezeitraums kann es jedoch nicht mehr gefunden werden, und die Frequenz fällt erneut auf einen Wert unterhalb von 0,5.

Für die intervallbasierte Heuristik wurden bei $k = 25$ Intervallen und einer minimalen absoluten Abweichung von $\epsilon = 1\%$ insgesamt 16 interessante Änderungen identifiziert, 12 Änderungen bzw. 75% davon waren in diesem Fall auch signifikant. Bei Verwendung der korridorbasierten Heuristik lag die Anzahl interessanter Änderungen mit 79 deutlich höher, wobei 22 signifikant waren. Es wird deutlich, daß der Anteil der signifikanten Änderungen, bezogen auf die Menge der interessanten Änderungen, für alle drei Heuristiken vergleichsweise hoch ist. Zumindest für den stabilitätsbasierten

Ansatz scheint das die im Abschnitt 5.3.4 geäußerte Vermutung zu bestätigen, wonach Änderungen, die die Sichtbarkeit eines Musters beeinflussen, häufig auch signifikant sind. Interessanterweise liefert die korridorbasierte Heuristik deutlich mehr interessante Änderungen als die intervallbasierte, obwohl sie eine sehr viel kleinere Anzahl von Fällen betrachtet. Das kann zum einen an der sehr kurzen Trainingsphase für die korridorbasierte Heuristik liegen: Nach nur 4 Perioden sind Durchschnitt und Standardabweichung noch nicht wohldefiniert, weswegen Verletzungen des Korridors sicher häufiger festgestellt werden können. Andererseits liegt dies jedoch auch daran, daß die Support-Zeitreihen der Muster nur sehr moderate Änderungen aufwiesen und ein einzelnes Intervall mit knapp 4% relativ breit war. Prinzipiell wäre es durchaus möglich, die Anzahl der Intervalle zu erhöhen, um mehr interessante Musteränderungen zu erhalten. Aber selbst bei $k = 35$ Intervallen konnten nicht wesentlich mehr interessante Änderungen gefunden werden. Als Ursache kommt dafür neben den bereits erwähnten moderaten Support-Schwankungen eine Unzulänglichkeit des intervallbasierten Ansatzes in Frage: Bei jeder Überschreitung einer Intervallgrenze wurde zusätzlich überprüft, ob die beobachtete Änderung größer oder gleich $\epsilon$ war, wobei in diesem Fall eine absolute Abweichung von $\epsilon = 1\%$ verwendet wurde. Bei einer Intervallbreite von 3,9% für $k = 25$ bzw. knapp 2,8% für $k = 35$ folgt daraus jedoch, daß Änderungen gleicher Größe in einem Fall zur Signalisierung einer interessanten Änderung führen und im anderen Fall nicht, je nachdem, ob der aktuelle Wert der Zeitreihe in der Nähe einer Grenze liegt.

Verglichen mit den Signifikanztests für alle Musteränderungen, finden die Heuristiken etwa die Hälfte aller signifikanten Änderungen, wobei jedoch eine unterschiedliche Anzahl von Fällen betrachtet wird. Würden die Signifikanztests ebenfalls erst ab der fünften Periode nach der ersten Sichtbarkeit eines Musters durchgeführt, sänke die Anzahl der gefundenen signifikanten Änderungen auf 30. Bei kombinierter Verwendung der Heuristiken können im Vergleich dazu insgesamt 25 und damit mehr als 80% aller signifikanten Änderungen gefunden werden. Berücksichtigt man die unterschiedlichen Definitionen von Interesse, die der Ermittlung interessanter Änderungen zugrunde liegen, überrascht dieses Ergebnis. Insbesondere die korridorbasierte Heuristik meldet jedoch noch sehr viel mehr interessante Änderungen. Unabhängig davon, welche Definition von Interesse verwendet wird, kann die Frage, welche dieser Änderungen tatsächlich interessant sind, nur durch eine genaue Betrachtung der entsprechenden Zeitreihe und unter Berücksichtigung des Anwendungskontextes beantwortet werden. In diesem Zusammen-

hang muß natürlich auch eine inhaltliche Würdigung der Muster erfolgen, und obwohl der durchgeführten Analyse ursprünglich keine anwendungsbezogene Zielstellung zugrunde lag, sollen die Implikationen, die sich aufgrund der beobachteten Änderungen ergeben, am Beispiel des Musters `I11` ⇒ `I3` verdeutlicht werden.

Der Inhalt des Musters besagt, daß Besucher der *Homepage* des Konzepts Information auch Übersichts- und Navigationsseiten aufsuchen. Während im betrachteten Fall zum Konzept `I11` nur eine einzelne Seite gehörte, beinhaltete das Konzept `I3` eine größere Anzahl von Seiten, z. B. Navigationsmenüs, die zu den verschiedenen Konzepten führten, aber auch die *Sitemap* und die Homepage des Servers. Tabelle 7.17 zeigt die Zeitreihen des Musters für den Support, die Konfidenz, den Lift und die absolute Anzahl der Sitzungen, die das Muster unterstützen. Das Muster erfüllt in allen Perioden die in der

| Periode | Support | Konfidenz | Lift | Anzahl |
|---|---|---|---|---|
| 1 | 0,116 | 0,843 | 1,08 | 295 |
| 2 | 0,129 | 0,859 | 1.12 | 335 |
| 3 | 0,113 | 0,876 | 1,14 | 204 |
| 4 | 0,115 | 0,870 | 1,19 | 140 |
| 5 | 0,090 | 0,822 | 1,10 | 263 |
| 6 | 0,108 | 0,853 | 1,12 | 295 |
| 7 | 0,076 | 0,837 | 1,05 | 174 |
| 8 | 0,065 | 0,848 | 1,18 | 196 |

Tabelle 7.17: Die verschiedenen Zeitreihen des Musters `I11` ⇒ `I3`

Mining-Anfrage festgelegten Kriterien und gehört somit zur Gruppe der *permanenten* Muster. Während die Konfidenz und der Lift mit Ausnahme des leicht erhöhten Niveaus in den Perioden 3 und 4 nur moderate Änderungen aufweisen, nimmt der Support des Musters zum Ende des Analysezeitraums deutlich ab.

Abbildung 7.9 zeigt die Ergebnisse der korridor- und intervallbasierten Heuristik. Die waagerechten Linien bezeichnen die Intervallgrenzen bei $k = 25$ Intervallen, wobei die untere Linie der zur Musterentdeckung verwendeten Support-Schwelle von $\tau_s = 2{,}5\%$ entspricht. Die senkrechte Linie bezeichnet das Ende der Lernphase für die korridorbasierte Heuristik. Es ist ersichtlich, daß die Werte der Zeitreihe im Vergleich zur jeweiligen Vorperiode in den Perioden 5, 6 und 7 in einem anderen Intervall liegen, wobei alle signalisierten

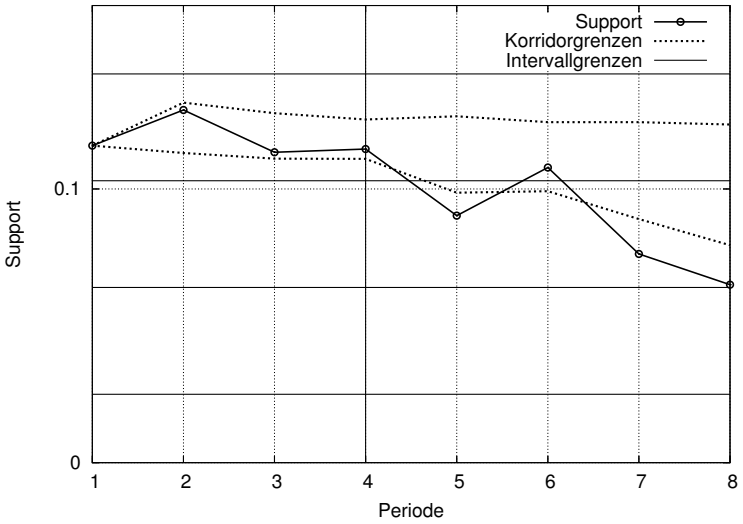


Abbildung 7.9: Die Entwicklung des Supports des Musters `I11` ⇒ `I3`

Änderungen auch signifikant waren. Die korridorbasierte Heuristik meldet in den Perioden 5 und 7 interessante Änderungen. Dem Nutzer wird zwar auch in Periode 8 signalisiert, daß der Support außerhalb des Korridors liegt, es wird jedoch kein weiterer Alarm ausgelöst, sondern die in Periode 7 signalisierte Änderung als langfristig gekennzeichnet. Dieses Beispiel zeigt einen wesentlichen Unterschied zwischen den Signifikanztests und der korridorbasierten Heuristik: Obwohl die Support-Änderung zwischen den Perioden 7 und 8 nicht signifikant war und gemäß diesem Kriterium keine interessante Änderung signalisiert wurde, weicht der beobachtete Wert nach wie vor von den Erwartungen des Nutzers ab und wird deshalb gemeldet.

Für den Anwender ist dieses Muster besonders interessant, weil es eines der wenigen ist, die einen über alle Perioden stabilen Zusammenhang in den Daten beschreiben. Er sieht außerdem, daß, obwohl das Muster mit fast unveränderter Konfidenz gilt, der Anteil der Sitzungen, in denen auf beide Konzepte gemeinsam zugegriffen wird, mit fortschreitender Analysedauer abnimmt. Damit zeigt das Muster einen zur Entwicklung der Gesamtzahl der Sitzungen entgegengesetzten Trend (vgl. Tabelle 7.13). Als Ursache für diese Entwicklung käme auch eine Verringerung der Anzahl jener Sitzungen in Frage, die nur eines der beiden Konzepte unterstützen. Um die beobachtete

Änderung bewerten zu können, ist es deswegen notwendig, sie detaillierter zu betrachten.

Tabelle 7.18 zeigt die Zeitreihen für den Support der beiden Konzepte. Es

| | I11 | | I3 | |
|---|---|---|---|---|
| Periode | Support | Anzahl | Support | Anzahl |
| 1 | 0,137 | 350 | 0,780 | 1986 |
| 2 | 0,150 | 390 | 0,766 | 1992 |
| 3 | 0,130 | 233 | 0,774 | 1392 |
| 4 | 0,132 | 161 | 0,733 | 896 |
| 5 | 0,110 | 320 | 0,746 | 2173 |
| 6 | 0,126 | 346 | 0,760 | 2078 |
| 7 | 0,091 | 208 | 0,796 | 1817 |
| 8 | 0,077 | 231 | 0,722 | 2175 |

Tabelle 7.18: Die Entwicklung der Einzelkonzepte

ist erkennbar, daß der Support des Konzepts `I11` eine ähnliche Entwicklung wie der des Musters aufweist. Nach einem kurzfristigen Anstieg in Periode 2 gibt es einen deutlichen Rückgang, der in Periode 6 kurz unterbrochen wird. Die Entwicklung des Konzepts `I3` zeigt dagegen einige Unterschiede. So fällt der Support in Periode 2 und steigt von Periode 4 bis 7 kontinuierlich an. Erst in Periode 8 geht der Support deutlich zurück. Interessanterweise fällt in Periode 8 der Support beider Konzepte, obwohl die Anzahl der Sitzungen, die die Konzepte unterstützen, in beiden Fällen ansteigt. Demnach wird der Anstieg der Gesamtzahl an Sitzungen, der in der letzten Periode zu beobachten ist, überwiegend von Besuchern hervorgerufen, welche diese beiden Konzepte seltener aufsuchen.

Natürlich kann nicht mit Sicherheit festgestellt werden, ob der Support-Rückgang des Musters auf den Support-Rückgang des Konzepts `I11` zurückzuführen ist – insbesondere in der letzten Periode verzeichnen beide Konzepte einen Rückgang des Supports. Aufgrund der Ähnlichkeit der Entwicklung des Supports des Musters und des Konzepts ist es jedoch zumindest sehr wahrscheinlich, daß es sich hier um eine der Ursachen dafür handelt. Für den Anwender hängt die Beurteilung der Änderung somit von der Bedeutung des Konzepts für das Erreichen der Zielstellungen ab, die er mit seinem Informationsangebot verfolgt. Da es sich in diesem Fall um die Homepage des

Informationsangebotes handelt, ist es möglich, daß sich die Änderung auch auf die Nachfrage nach den darunter liegenden Konzepten auswirkt, insbesondere wenn es nicht genügend Querverweise gibt. Sind die hier angebotenen Informationen wesentlich, muß der Betreiber versuchen, die Ursachen der Änderung zu ermitteln, die nicht aus der Protokolldatei hervorgehen, um ihnen durch geeignete Maßnahmen entgegenzuwirken.

### 7.2.3 Ermittlung der Ursachen von Musteränderungen

Wie gerade deutlich wurde, kann es für die Beurteilung von Musteränderungen hilfreich sein, sie im Zusammenhang mit den Änderungen ihrer Bestandteile zu betrachten. Ziel dabei ist es, die Änderungen ähnlicher Muster auf Änderungen gemeinsamer Elemente zurückzuführen und eine minimale Menge von Komponentenänderungen zu ermitteln, die als Ursache für alle beobachteten Musteränderungen in Frage kommen (vgl. Abschnitt 5.5). Für jedes Muster, das signifikante Änderungen gezeigt hat, wird dazu überprüft, ob eine seiner Komponenten ebenfalls signifikante Änderungen aufweist. In Periode 5 meldete z. B. die korridorbasierte Heuristik eine interessante Änderung für das Muster `I112`, `I113` ⇒ `I3`. Die Überprüfung der Bestandteile des Musters lieferte die in Tabelle 7.19 dargestellten Ergebnisse. Demnach änder-

| Komponente | signifikant |
|---|---|
| `I112` | – |
| `I112`, `I113` | √ |
| `I113` | – |
| `I113`, `I3` | √ |
| `I3` | – |
| `I3`, `I112` | – |

Tabelle 7.19: Überprüfung der Komponenten des Musters `I112`, `I113` ⇒ `I3`

te sich der Support nur für zwei der sechs Komponenten signifikant. Interessant an diesem Ergebnis ist, daß zwar für die Komponenten {`I112`, `I113`} und {`I113`, `I3`} signifikante Änderungen festgestellt werden konnten, nicht aber für die sie konstituierenden Subkomponenten. Das impliziert, daß es bezüglich der Zugriffe auf die einzelnen Konzepte keine wesentlichen Ände-

rungen gab, daß sich aber die Zusammensetzung der in einer Sitzung besuchten Konzepte geändert hat.

Tabelle 7.20 zeigt eine Zusammenfassung der Ergebnisse für die verschiedenen Heuristiken. Die zweite Spalte gibt die Anzahl der Komponentenände-

| | Komponentenänderungen | |
|---|---|---|
| Heuristik | betrachtet | signifikant |
| Signifikanztest (sichtbar) | 28 | 17 |
| Signifikanztest (beobachtet) | 59 | 28 |
| Signifikanztest (alle) | 226 | 102 |
| stabilitätsbasiert | 42 | 18 |
| intervallbasiert | 32 | 22 |
| korridorbasiert | 82 | 51 |

Tabelle 7.20: Komponentenänderungen für die verschiedenen Heuristiken

rungen an, die auf ihre Signifikanz untersucht wurden. Es ist ersichtlich, daß diese Zahl für die Signifikanztests, die auf der Menge aller Musteränderungen arbeiten, deutlich größer ist. Die Ursache dafür ist zum einen in der großen Anzahl signifikanter Änderungen zu sehen, die bei Einbeziehung aller Änderungen gefunden wurde. Zum anderen liegt dies jedoch auch daran, daß nur in diesem Fall signifikante Änderungen für Muster gefunden wurden, die mehr als drei Elemente enthielten. Gemäß dem in Abbildung 5.3 dargestellten Algorithmus werden zwar zunächst nur die längsten Komponenten eines Musters auf signifikante Änderungen untersucht, weisen diese jedoch ebenfalls signifikante Änderungen auf, werden auch sie in ihre Komponenten zerlegt und analysiert. Falls nun sehr viele Komponenten eines Musters signifikante Änderungen aufweisen, wächst die Zahl der zu überprüfenden Komponenten sehr stark an. In diesem Fall gab es allein 7 Muster der Länge 4, für die insgesamt 65 Komponenten untersucht wurden.

Ein weiteres interessantes Ergebnis ist darin zu sehen, daß ungeachtet der Frage, welche Heuristik verwendet wurde, jeweils nur bei rund der Hälfte der untersuchten Komponenten ebenfalls signifikante Änderungen zu verzeichnen waren.

Zweifellos stellt diese Analyse einen zusätzlichen, im Einzelfall eventuell auch sehr großen Aufwand dar. Ihren Ergebnissen lassen sich jedoch viele zusätzliche Informationen entnehmen, die eine bessere Bewertung der beob-

achteten Änderungen ermöglichen. Zum einen erhält der Anwender Hinweise, welches die möglichen Ursachen für eine beobachtete Musteränderung waren bzw. welche anderen Änderungen zur beobachteten beigetragen haben. Zum anderen erfährt der Anwender aber auch, welche Komponenten keine Änderungen zeigten und damit im betrachteten Zeitraum stabil waren. Unter Ausklammerung des nur bedingt realistischen Ansatzes, *alle* Musteränderungen auf ihre Signifikanz hin zu überprüfen, wurden über den gesamten Analysezeitraum 21 Muster gefunden, die in 35 Fällen signifikante Änderungen zeigten. Um die Ursachen für diese Änderungen zu ermitteln, mußten bei Durchführung offensichtlicher Optimierungen insgesamt nur 79 Komponenten analysiert werden, wovon 38 signifikante Änderungen aufwiesen.

### 7.2.4 Fazit

Welche Kriterien für die Bewertung von Musteränderungen verwendet werden, hängt letztlich vom Anwender bzw. vom jeweiligen Anwendungskontext ab. Bewertet der Anwender jedoch Musteränderungen ausschließlich aus statischer Sicht und betrachtet er nur die sichtbaren Muster, wird er mit großer Wahrscheinlichkeit wichtige Änderungen übersehen. Neben der absoluten oder relativen Stärke einer beobachteten Änderung ist für den Anwender vor allem interessant, welche Änderungen zum Erscheinen neuer Muster geführt haben bzw. welche Ursachen dafür in Frage kommen, daß ein aus früheren Perioden bekanntes Muster nicht mehr gefunden werden kann. Die im Abschnitt 5.3.4 vorgestellten Heuristiken beziehen jeweils unterschiedliche Aspekte der Stabilität von Mustern ein bzw. bewerten die beobachteten Änderungen aus verschiedenem Blickwinkel. Gemeinsam ist allen Heuristiken, daß sie – folgt man der jeweiligen Definition von Interesse – eine Menge interessanter Musteränderungen zurückliefern. Ob die signalisierten Änderungen für den Anwender aber wirklich interessant sind, ist im Rahmen des jeweiligen Anwendungsfalls zu entscheiden.

Auch wenn ein Vergleich der verschiedenen Methoden aus diesem Grund nicht sehr aussagekräftig erscheint, wurden die Ergebnisse der Heuristiken und der Signifikanztests gegenübergestellt. Es zeigte sich, daß die Menge der durch die Heuristiken gemeldeten Musteränderungen fast eine echte Obermenge der signifikanten Änderungen darstellt – nur einige wenige wurden nicht erkannt –, wenn die Methoden auf einer vergleichbaren Menge von Musteränderungen operieren. Interessiert sich der Anwender nur für jene Änderungen, die kein *Geräusch* darstellen, sind Signifikanztests sicher am besten

geeignet, die interessanten Musteränderungen zu identifizieren. Ist er darüber hinaus jedoch auch an Änderungen interessiert, die stärker von früheren Werten abweichen, als aufgrund seiner Erfahrungen aus der Vergangenheit anzunehmen wäre, stellt die korridorbasierte Heuristik – wie am Beispiel des Musters `I11` $\Rightarrow$ `I3` gezeigt wurde – eine sinnvolle Ergänzung der Analyse dar. Die detaillierte Betrachtung der Änderungen dieses Musters hat zudem gezeigt, welche zusätzlichen Erkenntnisse durch die Ausweitung der Betrachtung auf die Änderungen der Komponenten eines Musters gewonnen werden können.

Eine genaue Betrachtung der interessanten Änderungen, die durch die intervallbasierte Heuristik signalisiert wurden, offenbarte eine Ungleichbehandlung von Änderungen gleicher Stärke, die in Abhängigkeit vom jeweiligen Wert in einem Fall zur Signalisierung einer interessanten Änderung führten, im anderen Fall jedoch nicht. Um diesem Problem zu begegnen, müßte $\epsilon$ lediglich in Höhe der halben Intervallbreite gewählt werden. Dies wäre jedoch äquivalent zur Prüfung auf eine absolute Änderung größer oder gleich $\epsilon$. In diesem Fall wären eine Einteilung des Wertebereichs in Intervalle und die Prüfung, ob die Werte zweier aufeinanderfolgender Perioden in unterschiedliche Intervalle fallen, jedoch unnötig. Eine Ausnahme stellt der Fall dar, in dem die jeweiligen Intervallgrenzen im Rahmen des Anwendungskontextes von besonderem Interesse sind. Für ein solches Szenario würde dem Aufwand für die Bestimmung der Intervallgrenzen der zusätzliche Informationsgewinn, wenn sie überschritten werden, gegenüberstehen.

Wie im Abschnitt 5.3.4 ausgeführt wurde, reagiert die stabilitätsbasierte Gruppierung in frühen Phasen der Analyse sehr stark auf Änderungen der Sichtbarkeit eines Musters (vgl. Abbildung 7.8). In den Experimenten wurde deshalb eine Lernphase von 4 Perioden verwendet, bevor interessante Änderungen bezüglich der Frequenz eines Musters gemeldet wurden. Die gleiche Problematik tritt auch bei der korridorbasierten Heuristik auf. Während die Länge der Trainingsphase bei der stabilitätsbasierten Gruppierung noch als ausreichend angesehen werden kann, ist sie für die Berechnung des Korridors aus prinzipiellen Erwägungen wahrscheinlich zu kurz. Eine Möglichkeit, dieses Problem zu umgehen, bestünde in der Erhöhung der Anzahl der Perioden in der Lernphase. Für den untersuchten Datensatz hätte dies jedoch bedeutet, daß entweder interessante Änderungen nur in einem sehr kleinen Anteil der Perioden hätten gefunden werden können oder daß die Granularität der zeitlichen Dimension hätte erhöht werden müssen. Während die erste Alternative wahrscheinlich zu weniger aussagekräftigen Ergebnissen geführt hätte,

war die Anzahl der Sitzungen pro Monat bereits so gering, daß eine weitere Unterteilung der Zeitachse auf wöchentlicher Basis zu sehr instabilen Mustern geführt hätte.

## 7.3 Zusammenfassung

Das Ziel der beiden vorgestellten Fallstudien war es, die in den Kapiteln 4 und 5 beschriebenen Konzepte zur Erkennung und Bewertung von Musteränderungen zu überprüfen. In beiden Fällen wurden Transaktionsdaten untersucht, die über einen längeren Zeitraum gesammelt worden waren. Während das Hauptaugenmerk der ersten Studie jedoch Effizienzaspekten bei der Musterüberwachung galt, konzentrierte sich die zweite auf die Analyse der beobachteten Änderungen.

Im einzelnen befaßte sich der erste Fall mit der Fragestellung, ob es ausreicht, eine Teilmenge der gefundenen Muster zu überwachen, um auf interessante Änderungen der Regelbasis *und* anderer Muster zu schließen. Zu diesem Zweck wurde der Prozeß der Wissensentdeckung in zwei Phasen geteilt: die Mining-Phase, in der die Daten der ersten betrachteten Periode mit Hilfe konventioneller Mining-Techniken untersucht wurden, und die Monitoring-Phase, in der die Statistiken ausgewählter Muster in den folgenden Perioden überwacht wurden. Dabei sollten starke Änderungen beobachteter Muster als Indikator für Änderungen der Regelbasis und anderer, nichtüberwachter Muster dienen und die Notwendigkeit einer neuen Mining-Session signalisieren. Zum einen zeigte sich, daß die Überwachung einer Teilmenge der Muster nur unzureichende Rückschlüsse auf die Dynamik der Regelbasis zuläßt. Auch wenn dieses Ergebnis der verwendeten Arbeitshypothese widersprach, konnte es nicht überraschen und lag in der Auswahl zu überwachender Muster begründet: Häufige Muster weisen naturgemäß sehr wenige Änderungen auf, die ihre Sichtbarkeit beeinflussen. Gerade solche Änderungen beeinflussen die Dynamik der Regelbasis jedoch in erheblichem Maße. Auf der anderen Seite zeigten die Zeitreihen statistischer Eigenschaften überwachter und nichtüberwachter Muster deutliche Abhängigkeiten. Nur für einen sehr kleinen Teil der nichtüberwachten Muster – die darüber hinaus nur in einer Periode zutage traten – konnten keine erkennbaren Zusammenhänge gefunden werden. Außerdem wurde festgestellt, daß die Überwachung einer Teilmenge von Mustern sehr viel effizienter durchgeführt werden kann, wenn die Musterstatistiken direkt in der Datenbank bestimmt werden. Im Hinblick auf

einen subjektiven Ansatz zur Bestimmung interessanter Regeln könnte sich der Einsatz konventioneller Mining-Techniken dadurch völlig erübrigen.

Die zweite Fallstudie befaßte sich mit der Erkennung interessanter Musteränderungen, wofür die im Kapitel 5 beschriebenen Heuristiken eingesetzt wurden. Ein Vergleich der Anzahl an gefundenen Änderungen zeigte sich wenig aussagekräftig, da die verschiedenen Verfahren unterschiedliche Definitionen von Interesse verwenden. Um zu ermitteln, welche Definition im betrachteten Anwendungskontext am besten geeignet ist, müssen die Änderungen im Detail betrachtet werden. Dabei kann eine Unterscheidung kurz- und langfristiger Musteränderungen hilfreich sein. Für alle Muster, die eine interessante Änderung zeigten, wurden außerdem jene Bestandteile ermittelt, die ebenfalls interessante Änderungen aufwiesen und somit potentiell zur beobachteten Musteränderung beigetragen hatten. Obwohl diese Analyse einen zusätzlichen Aufwand darstellt, werden dadurch zumindest solche Ursachen für die beobachteten Musteränderungen aufgedeckt, die aus dem untersuchten Datensatz abzuleiten sind. Zusammen mit der zeitlichen Dimension der Änderungen bilden sie die Grundlage, um Musteränderungen in einen Bezug zu den Ereignissen zu setzen, die nicht aus den Daten hervorgehen.

# Kapitel 8

# Schlußfolgerungen

Die Einsicht, daß ein tieferes Verständnis für die in den Daten enthaltenen Zusammenhänge einen wesentlichen Wettbewerbsvorteil darstellen kann, und nicht zuletzt die technologische Entwicklung der letzten Jahre haben zu einem enormen Wachstum hinsichtlich der Anzahl und Größe der gesammelten Datensätze geführt und die Entwicklung von Methoden für ihre effiziente Analyse zu einer großen Herausforderung werden lassen. Während die erfaßten Daten im Rahmen der Musterentdeckung traditionell als statische Einheit betrachtet wurden, hat sich inzwischen eine dynamische Sichtweise etabliert. Dieser Ansatz trägt der Tatsache Rechnung, daß ein Datensatz im Normalfall über einen bestimmten Zeitraum gesammelt wurde und sich ändernde Zusammenhänge enthalten kann.

In den ersten Arbeiten, die sich mit dieser Problematik beschäftigten, wurden deshalb Methoden zur Aktualisierung der gefundenen Muster vorgeschlagen, wobei häufig Effizienzüberlegungen im Vordergrund standen (vgl. Abschnitt 3.1). Ein wesentlicher Nachteil dieser Beiträge ist jedoch in bezug auf den Umfang der analysierten Daten zu erkennen: Inkrementelle Techniken betrachten grundsätzlich die *Gesamtheit* der bis zum jeweiligen Analysezeitpunkt erfaßten Daten. Da somit alle über den bisherigen Erfassungszeitraum gesammelten Daten aggregiert werden, reflektieren die aktualisierten Muster nur scheinbar die gegenwärtigen Zusammenhänge. Darüber hinaus können auch keine Aussagen hinsichtlich der *Entwicklung* der Muster im Zeitverlauf gemacht werden. Die im Abschnitt 4.3.1 beschriebene inkrementelle Bestimmung der Musterstatistiken behebt das Problem nur teilweise. Zwar erhält der Anwendungsexperte in diesem Fall Zeitreihen für die statistischen Eigenschaften der Muster, er kann ihre Entwicklung jedoch nur eingeschränkt

nachvollziehen, da die Daten noch immer in der für einen solchen Ansatz typischen, stark aggregierten Form vorliegen. Auch die im Abschnitt 3.4 vorgestellte graduelle Anpassung eines Klassifikators unter Verwendung eines gleitenden Zeitfensters kann das Problem nicht lösen: Für den Anwender ist es besonders wichtig, die aufgetretenen Änderungen nachvollziehen zu können, da viele Ursachen, die Veränderungen in den Daten hervorrufen, nicht im analysierten Datensatz erfaßt sind. Will er Bezüge zwischen den Ereignissen, die möglicherweise einen Einfluß ausüben, und den beobachteten Änderungen der gefundenen Muster herstellen, muß der Anwender in der Lage sein, die Musteränderungen eindeutigen Zeitpunkten zuzuordnen. Weder die inkrementelle Aktualisierung der Muster noch das graduelle Lernen auf Basis eines Zeitfensters können jedoch diese Art der Information liefern.

Aus diesen Gründen wird in der vorliegenden Arbeit statt der Aktualisierung die *Fortschreibung* der gefundenen Muster vorgeschlagen. Dazu werden in bestimmten Abständen Mining-Sessions durchgeführt, in denen jeweils die seit der letzten Sitzung gesammelten Daten analysiert werden. Im Ergebnis liegt für jede Session eine Menge von Mustern vor, von denen ein Teil bereits aus früheren Sitzungen bekannt ist, ein anderer Teil jedoch neu sein kann. Zusätzlich könnte der Fall eintreten, daß die Muster, die in früheren Perioden gefunden wurden, nicht mehr in den Ergebnissen erscheinen. Alle in einer Mining-Session gefundenen Muster werden in die Regelbasis eingetragen, oder es werden – sofern die Muster bereits gespeichert sind – ihre mit dem Zeitstempel der aktuellen Mining-Session versehenen statistischen Eigenschaften erfaßt. Die Grundlage hierfür bildet das im Kapitel 4 vorgestellte temporale Regelmodell. Durch seine Verwendung ist die Erkennung von Änderungen bekannter Muster prinzipiell unproblematisch, da sie unmittelbar aus den Zeitreihen ihrer statistischen Eigenschaften hervorgehen. Neue bzw. nicht mehr erscheinende Muster können dagegen durch einen Abgleich mit den bekannten Mustern ermittelt werden.

Aus der Sicht des Nutzers ergeben sich damit zwei verschiedenartige Ebenen von Musteränderungen, die sich bezüglich ihres Einflusses auf die Sichtbarkeit eines Musters unterscheiden. Auf der einen Seite gibt es Änderungen der statistischen Eigenschaften einer Regel, welche ihre Sichtbarkeit nicht beeinflussen, d. h. die festgelegten Schwellen für z. B. Support und Konfidenz nicht unterschreiten. Auf der anderen Seite stehen Änderungen, die Einfluß auf die Sichtbarkeit eines Musters nehmen, weil sie die in der Mining-Anfrage definierten Kriterien verletzen, sie zum ersten Mal oder wieder erfüllen. Der zweite Typ von Änderungen hat somit direkten Einfluß auf die Regelbasis

und ist für den Nutzer im allgemeinen von besonderem Interesse. Die Verwendung der im Abschnitt 5.2.1 gegebenen Definitionen unterschiedlicher Arten von Musteränderungen versetzt den Anwender in die Lage, zwischen diesen beiden Ebenen zu unterscheiden.

Welche Art von Änderungen häufiger zu beobachten sein wird, hängt im wesentlichen davon ab, ob die Werte der Zeitreihen in der Nähe der in der Mining-Anfrage definierten Schwellen liegen und welchen Schwankungen sie ausgesetzt sind. In den Fallstudien zeigte sich, daß die Änderungen auf Musterebene den Großteil der beobachteten Änderungen ausmachen und nur eine relativ kleine Anzahl die Regelbasis betraf. Dieses Ergebnis kann jedoch kaum überraschen, da für die sichtbaren Muster in jeder Periode Änderungen der statistischen Eigenschaften zu verzeichnen sein werden. Aus diesem Grund muß die Anzahl der gemeldeten Änderungen in geeigneter Weise reduziert werden. Neben den verschiedenen subjektiven und objektiven Kriterien, die in den Abschnitten 5.3.2 und 5.3.3 beschrieben sind, wurden zu diesem Zweck die im Abschnitt 5.3.4 vorgestellten Heuristiken entwickelt. Sie erlauben dem Anwendungsexperten, die für ihn interessanten Änderungen zu ermitteln, wobei die Relevanz einer Änderung durch die Betrachtung unterschiedlicher Aspekte der *Stabilität* von Mustern bestimmt wird.

Änderungen auf der Ebene der Regelbasis sind für den Anwender immer dann interessant, wenn sie nicht nur durch marginale Änderungen der statistischen Eigenschaften hervorgerufen werden. Im allgemeinen kann der Nutzer mit Hilfe konventioneller Mining-Techniken jedoch nicht die Stärke solcher Änderungen ermitteln. Auch ein Abgleich mit in früheren Perioden gefundenen Mustern ist oft bloß eingeschränkt möglich, da nur wenige Mining-Algorithmen ihre Ergebnisse in strukturierter Form speichern. Beide Probleme lassen sich durch den im Kapitel 6 vorgestellten Pattern Monitor beheben. Basierend auf dem temporalen Regelmodell, speichert er alle in der Vergangenheit gefundenen Muster, wodurch ein Vergleich der Muster der aktuellen Periode mit jenen früherer Perioden stark vereinfacht wird. In Verbindung mit der SQL-basierten Ermittlung der statistischen Eigenschaften von Mustern ist er darüber hinaus in der Lage, auch das Ausmaß der Änderungen zu bestimmen, die zu Veränderungen in der Regelbasis geführt haben (vgl. Abschnitt 4.3.2). Zusätzlich lassen sich mit Hilfe der verschiedenen PAM-Workflows völlig unterschiedliche Informationsbedürfnisse befriedigen.

Die Quintessenz aller im Rahmen dieser Arbeit dargelegten Vorschläge besteht darin, den Anwender so detailliert wie möglich über die Zusammenhänge und Beziehungen in den untersuchten Daten zu informieren. Um

dieses Ziel zu erreichen, genügt es jedoch nicht, nur die vorhandenen Muster zu ermitteln. Vielmehr sollten auch jene Erkenntnisse berücksichtigt werden, die sich aus der Entwicklung der Muster über die Zeit ergeben können. So ist es nicht unwahrscheinlich, daß sich z. B. die Sichtbarkeit eines Musters periodisch ändert. Für den Anwender mag das Muster selbst bereits interessant sein, die Information, daß es nur in bestimmten Perioden in den Daten gefunden werden kann, stellt jedoch in jedem Fall einen besonderen Umstand dar und könnte für den betrachteten Anwendungskontext von zusätzlicher Wichtigkeit sein. Dieser Aspekt gewinnt an Bedeutung, wenn das Muster einen Zusammenhang repräsentiert, der im Rahmen der Anwendung zu erwarten und somit nicht überraschend wäre. Im Normalfall würde der Anwender einem solchen Muster sicher wenig Beachtung beimessen. Zeigt das Muster hingegen die beschriebene Periodizität, weckt es vielleicht doch sein Interesse. Aus dem beschriebenen Sachverhalt erwächst die Notwendigkeit, die Kriterien zur Bewertung der Wichtigkeit eines Musters zu erweitern und auf seine temporalen Eigenschaften auszudehnen. Eine unabdingbare Voraussetzung dafür ist jedoch, daß diese Dimension des entdeckten Wissens auch erfaßt wird.

Es wird offenbar, daß die Berücksichtigung der *Evolution* des entdeckten Wissens weitere und bedeutsame Erkenntnisse hinsichtlich des durch die erfaßten Daten abgebildeten Ausschnitts der Realität liefern kann. Der Prozeß der Überwachung und Analyse der gefundenen Änderungen hat dabei viele unterschiedliche Facetten. Zum einen sind technische Probleme wie die Ermittlung und Speicherung der Muster und ihrer statistischen Eigenschaften zu lösen. Zum anderen muß jedoch auch zwischen den verschiedenen Typen von möglichen Änderungen differenziert werden und eine Bewertung der festgestellten Änderungen vorgenommen werden. In der vorliegenden Arbeit wurden für die meisten der angesprochenen Problemstellungen Lösungsvorschläge erarbeitet. Darüber hinaus wurden die unterschiedlichen Aspekte der Entwicklung von Regeln in einem allgemeinen Bezugssystem erfaßt und der Prototyp eines Pattern Monitor entwickelt, der den Anwender über den gesamten, *erweiterten* Prozeß der Wissensentdeckung hinweg begleitet. Um Entscheidungsprozesse dauerhaft und wirksam unterstützen zu können, muß die Validität der gewonnenen Erkenntnisse kontinuierlich überprüft werden. Wie sich in den Fallstudien zeigte, werden die zur Pflege des entdeckten Wissens entwickelten Techniken diesem Anspruch gerecht. Zusätzlich wird der Anwender bei der Analyse der beobachteten Änderungen unterstützt. Auf der einen Seite stehen dabei die Heuristiken, welche die potentiell große Anzahl

gemeldeter Änderungen vermindern, indem sie die Menge von Änderungen bestimmen, die der jeweils verwendeten Definition von *Interesse* genügt. Auf der anderen Seite steht die Ermittlung der Ursachen von Musteränderungen, bei der für jedes Muster bestimmt wird, welche seiner Bestandteile zur beobachteten Änderung beigetragen haben. Der Anwendungsexperte verfügt somit über alle notwendigen Informationen, die sich unmittelbar aus den gesammelten Daten ergeben und die ihm eine subjektive, anwendungsspezifische Bewertung der Muster gestatten. Mit diesem Wissen versehen, ist er unter Umständen in der Lage, jene auslösenden Faktoren zu ermitteln, die sich *nicht* aus den Daten ableiten lassen.

# Kapitel 9

# Ausblick

Die Probleme, die bei der Beobachtung der Evolution von Mustern zutage treten, sind vielschichtiger Natur. Ein Faktor, der die Komplexität einer solchen Betrachtung wesentlich erhöht, sind die vielen verschiedenen Formen, in denen sich das entdeckte Wissen präsentiert. In dieser Arbeit lag der Schwerpunkt vor allem auf der Überwachung von Assoziationsregeln, und wenn auch die Verwendung einer Warenkorbanalyse durch eine entsprechende Problem-Modellierung häufig möglich sein wird, so müssen die vorgestellten Konzepte letztlich auf andere Mining-Paradigmen übertragen werden. Die Frage der strukturierten Erfassung der Analyse-Ergebnisse anderer Methoden der Wissensentdeckung wurde bereits beim Entwurf des temporalen Regelmodells berücksichtigt, und auch für die Überwachung der Evolution der Ergebnisse einer Cluster-Analyse wurde eine Methode entwickelt. Neben einer praktischen Überprüfung dieser Vorschläge sind jedoch auch Techniken notwendig, welche die *Beobachtung* anderer Mining-Ergebnisse erlauben. Die Überwachung der statistischen Eigenschaften wird in den meisten Fällen unproblematisch sein. Schwieriger ist die Einbeziehung von inhaltlichen Änderungen. Der im Abschnitt 5.2.1 vorgestellte Lösungsansatz widmet sich zwar diesem Problem, betrachtet jedoch nur die Ergebnisse einer Warenkorbanalyse. Für andere Paradigmen bleibt diese Frage zur Zeit weitestgehend unbeantwortet. Die Notwendigkeit für eine Übertragung verstärkt sich jedoch angesichts der Tatsache, daß bei bestimmten Ergebnissen eine bloße Überwachung der Musterstatistiken nicht aussagekräftig genug wäre. Als Beispiel seien hier Entscheidungsbaum-Klassifikatoren genannt: So hätte jede Änderung des Entscheidungsbaumes bei Anwendung der im Abschnitt 4.2.1 beschriebenen Methode zur Ermittlung der Musterkennung ein neues Muster zur

Folge, was in diesem Kontext sicher nicht wünschenswert wäre. Eine mögliche Lösung läge in der Verwendung eines geeigneten Abstandsmaßes, das – z. B. auf der Basis des in [Wang u. a. 1998] gemachten Vorschlags – die Ähnlichkeit zwischen zwei Entscheidungsbäumen angibt. Solange der Abstand eine vom Nutzer vorgegebene Grenze $\tau_{dist}$ nicht überschreitet, würde es sich demnach um inhaltliche Änderungen des Klassifikators handeln.

Obwohl dem Anwender im Resultat aller Analysen detaillierte Informationen über die in den Daten enthaltenen Zusammenhänge vorliegen, wird die Ermittlung der Ursachen für Musteränderungen, die nicht aus den gesammelten Daten hervorgehen, keine leicht zu lösende Aufgabe sein. Sicher ist es möglich, anhand der temporalen Entwicklung der Muster und entsprechend den Zeitpunkten externer Ereignisse Rückschlüsse auf mögliche Ursachen für die beobachteten Änderungen zu ziehen, es treten jedoch einige praktische Probleme auf wie z. B. die Erfassung der externen Einflüsse. So konnte im Rahmen der im Abschnitt 7.1 beschriebenen Analyse von Kommunikationsmustern ein extremer Anstieg der internen Kommunikation im Monat Februar festgestellt werden. Wer mit dem Universitätsbetrieb vertraut ist, wird bemerken, daß die Zunahme mit dem Ende der Vorlesungszeit und dem Beginn der Prüfungsperiode zusammenfällt – dem Zeitraum, in welchem normalerweise die Klausuren vorbereitet werden. Vor diesem Hintergrund wäre ein erhöhter Kommunikationsbedarf innerhalb des Instituts durchaus erklärbar. Es wird jedoch deutlich, daß dieser Schluß nur durch die Kenntnis des Anwendungskontextes möglich wird und es eigentlich keine Evidenz für einen kausalen Zusammenhang gibt. Die Unsicherheit eines solchen Schlusses ließe sich prinzipiell durch Verwendung von Methoden, wie sie im Abschnitt 3.5 vorgestellt wurden, beheben. Häufig liegt diese Art von externen Daten aber nicht vor bzw. wird nicht in geeigneter Weise erfaßt. Eine lohnende Herausforderung bestünde deshalb in der Entwicklung formaler Methoden, die zusätzliche Informationen dieses Typs in die Analyse einbeziehen können.

Ein weiteres Problem, für das es noch keine allgemeingültige Lösung gibt und das z. B. auch bei der Analyse von kontinuierlichen Datenströmen gelöst werden muß, ist die *richtige* Partitionierung der Daten. In den im Kapitel 7 beschriebenen Fallstudien wurde der Datensatz einfach auf der Basis einer Woche bzw. eines Monats geteilt. Bei dieser Herangehensweise besteht jedoch im allgemeinen die Gefahr, daß Muster übersehen werden. Die in [Chakrabarti u. a. 1998] vorgeschlagene Methode hatte den Nachteil, daß die Partitionierung von den jeweils betrachteten Regeln abhängt (vgl. Abschnitt 3.3.2). Auch bei der Ermittlung von Gültigkeitsintervallen, wie sie

in [Chen und Petrounias 1999] oder [Chang u. a. 2002] vorgenommen wurde, wird die Zeitachse auf der Basis individueller Muster geteilt. Eine von den Mustern unabhängige Herangehensweise bestünde darin, immer ein vergleichbares Datenvolumen zu analysieren oder die Granularität der Zeitachse zu variieren. Die Entwicklung von Techniken, welche die optimale Größe einer Partition losgelöst vom Anwendungskontext bestimmen können, wäre für die Überwachung der Evolution von Mustern ein wichtiger Beitrag.

Hinsichtlich der Fallstudien ist anzumerken, daß jeweils nur eine relativ kleine Anzahl unterschiedlicher Items in den Datensätzen gefunden werden konnte. Auch wenn dies die Qualität der Analysen nicht einschränkt, müßte überprüft werden, in welchem Maße die erhaltenen Ergebnisse auf andere Domänen übertragen werden können. Wünschenswert wäre deshalb eine Untersuchung, wie sich Datensätze mit unterschiedlichen Eigenschaften, z. B. einer größeren Menge an verschiedenen Items, auf die Analyse-Ergebnisse auswirken. Für diesen Zweck könnte auch ein synthetischer Datensatz Verwendung finden, dessen Eigenschaften zum Zwecke der Simulation verschiedener Perioden in geeigneter Weise variiert werden. In diesem Zusammenhang wäre auch von Interesse, wie sich eine höhere Anzahl von Items auf die Performanz der SQL-basierten Aktualisierung der Muster auswirkt. Um die praktische Eignung des entwickelten Pattern Monitors zu verifizieren, könnte zudem eine empirische Studie durchgeführt werden, die der Frage nachgeht, ob die zusätzlichen Informationen über die Evolution der gefundenen Muster in der Praxis wirklich relevant sind.

# Verzeichnis verwendeter Symbole

| | |
|---|---|
| $a$ | Klassifikationsgenauigkeit |
| $\alpha$ | Konfidenzniveau |
| $A_j$ | partiell geordnete Menge von Ereignissen |
| $\beta, \gamma$ | häufige Sequenzen in $S$ |
| $c$ | Konfidenz |
| $c(X \Rightarrow Y)$ | Konfidenz der Assoziationsregel $X \Rightarrow Y$ |
| $cf(X \Rightarrow Y)$ | Certainty Factor der Assoziationsregel $X \Rightarrow Y$ |
| $comp(\xi)$ | Menge der Komponenten des Muster $\xi$ |
| $\|comp(\xi)\|$ | Anzahl der Komponenten des Musters $\xi$ |
| $\|comp_{longest}(\xi)\|$ | Anzahl der längsten Komponenten des Musters $\xi$ |
| $C_i$ | Cluster $i$ |
| $dist$ | Distanzfunktion |
| $dist(o_i, o_j)$ | Distanz zwischen den Objekten $o_i$ und $o_j$ |
| $\delta$ | Aktualisierungen |
| $\|\delta\|$ | Anzahl der Transaktionen in $\delta$ |
| $D$ | Datensatz bzw. Datenbank |
| $D_i$ | Datensatz der Periode $i$ |
| $D_{test}$ | Testdaten |
| $D_{training}$ | Trainingsdaten |

| | |
|---|---|
| $D + \delta$ | aktualisierte Datenbank |
| $\|D\|$ | Anzahl der Transaktionen in $D$ |
| $e$ | Klassifikationsfehler |
| $e_a$ | beobachteter Klassifikationsfehler |
| $ec(X \Rightarrow Y)$ | erwartete Konfidenz der Assoziationsregel $X \Rightarrow Y$ |
| $\epsilon$ | Minimum der absoluten Änderung |
| $E$ | Menge von Ereignissen |
| $g_{ij}$ | Grenze zwischen den Clustern $C_i$ und $C_j$ |
| $i_i$ | Literal $i$ |
| $i, j$ | Indizes, $i, j \in \mathbf{N}$ |
| $I$ | Menge der Literale |
| $k$ | Anzahl der Iterationen bzw. Items, $k \in \mathbf{N}$ |
| $K_i$ | Klasse $i$ |
| $K(C_i)$ | Kern des Cluster $C_i$ |
| $l(X \Rightarrow Y)$ | Lift der Assoziationsregel $X \Rightarrow Y$ |
| $L_k^D, L_k^{D+\delta}$ | häufige $k$-Itemsets in $D$ bzw. $D + \delta$ |
| $m_i$ | Mittelpunkt des Clusters $C_i$ |
| $\mu$ | Mittelwert |
| $NB(L_k^D), NB(L_k^{D+\delta})$ | Negative Border von $L_k^D$ bzw. $L_k^{D+\delta}$ |
| $o_i$ | Objekt $i$ |
| $O$ | Objektraum |
| $O_{neg}$ | Menge der falsch klassifizierten Objekte |
| $O_{pos}$ | Menge der korrekt klassifizierten Objekte |
| $p_i$ | statistische Eigenschaft $i$ |
| $P$ | Menge von Mustern |
| $s$ | Support |
| $s(X)$ | Support des Itemsets $X$ |
| $s(X \Rightarrow Y)$ | Support der Assoziationsregel $X \Rightarrow Y$ |

| | |
|---|---|
| $s_i(\xi)$ | Support des Musters $\xi$ in Periode $i$ |
| $\mathsf{s}$ | Ereignis-Sequenz |
| $sim$ | Ähnlichkeitsfunktion |
| $S$ | Menge von Sequenzen bzw. Sequenzdatenbank |
| $\sigma$ | Standardabweichung |
| $\tau_c$ | Schwelle für die Konfidenz |
| $\tau_s$ | Schwelle für den Support |
| $T$ | Menge von Transaktionen |
| $TD(C), TD^2(C)$ | Kompaktheit des Clusters $C$ |
| $t_i$ | Zeitpunkt bzw. Periode $i$ |
| $T_E$ | Endzeitpunkt einer Sequenz |
| $T_S$ | Startzeitpunkt einer Sequenz |
| $\xi$ | Muster |
| $\|X\|$ | Anzahl der Transaktionen, die das Itemset $X$ enthalten |
| $X, Y$ | Mengen von Items, $X, Y \in I$ |
| $X \Rightarrow Y$ | Assoziationsregel zwischen den Itemsets $X$ und $Y$ |

# Glossar

| | |
|---|---|
| *Apriori* | ein Algorithmus für die Warenkorbanalyse (vgl. Abschnitt 2.2.1) |
| *Assoziationsregel* | das Ergebnis einer Warenkorbanalyse; eine Implikation der Form „*Wenn Voraussetzung, dann Ergebnis*“, wobei Voraussetzung und Ergebnis Itemsets sind (vgl. Abschnitt 2.3) |
| *azyklischer Graph* | eine durch gerichtete Kanten verbundene Knotenmenge, die keine Zyklen aufweist |
| *Centroid* | der errechnete Mittelpunkt einer Menge von Objekten (vgl. Abschnitt 2.2.3) |
| *Datenmodell* | die Festlegung der generischen Strukturen und Operatoren, die zur Modellierung einer bestimmten Anwendung genutzt werden können [Kemper und Eickler 1997] |
| *elementare Entität* | unteilbares, wohlunterscheidbares physisches oder gedankliches Konzept [Kemper und Eickler 1997] |
| *Frequent Itemset* | eine Menge von *Items*, die das vorgegebene Kriterium für den Support $\tau_s$ erfüllt und somit *häufig* ist (vgl. Abschnitt 2.2.1) |
| *Information Retrieval* | ein Forschungsgebiet der Informatik, das sich mit der Entwicklung von Methoden zur inhaltlichen Suche in unstrukturierten Daten, wie z. B. Textdokumente, beschäftigt |

| | |
|---|---|
| *Internet Service Provider* | ein Dienstleister, der seinen Kunden den Zugriff auf das Internet und/oder die Präsenz im Internet ermöglicht |
| *Item* | engl. Element, Gegenstand; Betrachtungseinheiten bzw. Entitäten, z. B. Produkte, die zum Zwecke einer Warenkorbanalyse zu sogenannten Transaktionen zusammengefaßt werden (vgl. Abschnitt 2.2.1) |
| *Konfidenz* | eine Maßzahl, welche die Validität der gefundenen Zusammenhänge ausdrückt (vgl. Abschnitt 2.4) |
| *Literal* | eine konstante alphanumerische Zeichenkette |
| *Medoid* | ein repräsentativer Vertreter einer Menge von Objekten (vgl. Abschnitt 2.2.3) |
| *Metrik* | eine Funktion, die den Abstand zwischen je zwei Elementen einer Menge definiert (vgl. Abschnitt 2.2.3) |
| *Muster* | die grundlegende Abstraktion zur Darstellung der in den Daten gefundenen Zusammenhänge (vgl. Abschnitt 2.3) |
| *Pattern Matching* | ein Forschungsgebiet der Informatik, das sich mit der Entwicklung von Methoden für die Suche nach (vorgegebenen) Mustern in unstrukturierten Daten, wie z. B. Textdokumente, beschäftigt |
| *Regel* | Synonym für *Muster* |
| *Regelbasis* | eine Datenbank, welche die Gesamtheit der gefundenen Muster speichert (vgl. Kapitel 4) |
| *Sequenz* | eine geordnete Menge von Objekten |
| *Supervised Learning* | Oberbegriff für Mining-Techniken, die eine Menge bekannter Instanzen zur Ableitung eines Modells verwenden, mit dessen Hilfe Aussagen über unbekannte Instanzen gemacht werden können (vgl. Abschnitt 2.2) |

| | |
|---|---|
| *Support* | eine Maßzahl, die den relativen Anteil der Datensätze angibt, die den gefundenen Zusammenhang unterstützen (vgl. Abschnitt 2.4) |
| *Transaktion* | eine Menge von Operationen oder Objekten, die zu einer logischen Einheit zusammengefaßt sind |
| *Unsupervised Learning* | Oberbegriff für Mining-Techniken, die eine Menge unbekannter Instanzen verwenden, um ein Modell abzuleiten, das die in den Daten enthaltenen Zusammenhänge beschreibt und eine bestimmte, vordefinierte Evidenz aufweist (vgl. Abschnitt 2.2) |
| *Wissensentdeckung* | der Prozeß der Extraktion von Wissen in Form von Mustern aus den erfaßten Daten (vgl. Abschnitt 2.1) |
| *Workflow* | engl. Arbeitsablauf; die für eine bestimmte Tätigkeit oder zur Lösung einer bestimmten Aufgabe vorgeschlagene bzw. festgelegte Handlungsabfolge |
| *Zeitstempel* | Attribut bzw. Feld, das Uhrzeit und/oder Datum des betrachteten Objekts speichert |

# Danksagung

Auf die eine oder andere Weise waren sehr viele Menschen an der Entstehung dieser Arbeit beteiligt. Da eine Liste der Personen, die mich unterstützt haben, mit großer Wahrscheinlichkeit unvollständig wäre, möchte ich mich an dieser Stelle bei allen bedanken, die mir im Verlauf der Dissertation geholfen haben.

Mein besonderer Dank gilt Prof. Myra Spiliopoulou und Prof. Oliver Günther, die mich in den vergangenen fünf Jahren unterstützt haben. Während Prof. Spiliopoulou mir am Anfang vor allem dabei half, mich thematisch zu orientieren, hat Prof. Günther mir die für die Realisierung der Arbeit notwendige Freiheit gewährt.

Meiner Frau Judith danke ich vor allem für ihre schier unerschöpfliche Geduld und Kraft, mit der sie es mir in den zurückliegenden sieben Monaten ermöglicht hat, mich auf die Fertigstellung der Dissertation zu konzentrieren.

Ich möchte mich bei Herbert Klemt und seiner Frau Angela bedanken, die dafür gesorgt haben, daß der Text – trotz der für einen Außenstehenden wohl eher trockenen Materie – lesbar geblieben ist.

Die im Vergleich mehr als faire „Asylpolitik“ von Gerrit Riessen bildete die Grundlage für die wohl produktivste Phase beim Verfassen dieser Arbeit. Außerdem möchte ich mich bei Matthias Fischmann bedanken, der sich in den letzten Wochen als geduldiger Zuhörer erwiesen hat.

# Literaturverzeichnis


Agrawal u. a. 1993 Agrawal, Rakesh ; Imielinski, Tomasz ; Swami, Arun N.: Mining Association Rules between Sets of Items in Large Databases. In: Buneman, Peter (Hrsg.) ; Jajodia, Sushil (Hrsg.): *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, ACM Press, 1993, S. 207–216

Agrawal u. a. 1995 Agrawal, Rakesh ; Psaila, Giuseppe ; Wimmers, Edward L. ; Zaït, Mohamed: Querying Shapes of Histories. In: *Proceedings of the 21st International Conference on Very Large Data Bases*. Zürich, Switzerland : Morgan Kaufmann, 1995, S. 502–514

Agrawal und Srikant 1994 Agrawal, Rakesh ; Srikant, Ramakrishnan: Fast Algorithms for Mining Association Rules in Large Databases. In: Bocca, Jorge B. (Hrsg.) ; Jarke, Matthias (Hrsg.) ; Zaniolo, Carlo (Hrsg.): *VLDB'94, Proceedings of the 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, Morgan Kaufmann, 1994, S. 487–499. – ISBN 1-55860-153-8

Agrawal und Srikant 1995 Agrawal, Rakesh ; Srikant, Ramakrishnan: Mining Sequential Patterns. In: *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*. Tapei, Taiwan : IEEE Computer Society, March 1995, S. 3–14

Aslam u. a. 1999 Aslam, Javed ; Pelekhov, Katya ; Rus, Daniela: A Practical Clustering Algorithm for Static and Dynamic Information Organization. In: *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, January 1999, S. 51–60

Au und Chan 2002 Au, Wai-Ho ; Chan, Keith C. C.: Fuzzy Data Mining

for Discovering Changes in Association Rules over Time. In: *Proceedings of the 11th IEEE International Conference on Fuzzy Systems*. Honolulu, USA : IEEE Computer Society, May 2002, S. 890–895

Ayan u. a. 1999 Ayan, Necip F. ; Tansel, Abdullah U. ; Arkun, Erol: An Efficient Algorithm To Update Large Itemsets With Early Pruning. In: *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego, CA, USA : ACM Press, August 1999, S. 287–291

Baron und Spiliopoulou 2001 Baron, Steffan ; Spiliopoulou, Myra: Monitoring Change in Mining Results. In: *Proceedings of the 3rd International Conference on Data Warehousing and Knowledge Discovery*. Munich, Germany : Springer, September 2001 (LNCS), S. 51–60

Baron und Spiliopoulou 2002 Baron, Steffan ; Spiliopoulou, Myra: Monitoring the Results of the KDD Process: An Overview of Pattern Evolution. In: Meij, Jeroen (Hrsg.): *Dealing with the data flood: mining data, text and multimedia*. The Hague, Netherlands : STT Netherlands Study Center for Technology Trends, April 2002, Kap. 6, S. 845–863

Baron und Spiliopoulou 2003 Baron, Steffan ; Spiliopoulou, Myra: Monitoring the Evolution of Web Usage Patterns. In: *1st European Web Mining Forum at ECML/PKDD 2003*. Cavtat-Dubrovnik, Croatia, September 2003, S. 64–79

Baron u. a. 2003 Baron, Steffan ; Spiliopoulou, Myra ; Günther, Oliver: Efficient Monitoring of Patterns in Data Mining Environments. In: *7th East-European Conference on Advance in Databases and Information Systems (ADBIS'03)*. Dresden, Germany : Springer, September 2003 (LNCS), S. 253–265

Berry und Linoff 1997 Berry, Michael J. ; Linoff, Gordon: *Data Mining Techniques: For Marketing, Sales and Customer Support*. John Wiley & Sons, Inc., 1997

Case u. a. 2001 Case, John ; Jain, Sanjay ; Kaufmann, Susanne ; Sharma, Arun ; Stephan, Frank: Predictive Learning Models for Concept Drift. In: *Theoretical Computer Science* 268 (2001), October, Nr. 2, S. 323–349. – ISSN 0304-3975

Chakrabarti u. a. 1998 Chakrabarti, Soumen ; Sarawagi, Sunita ; Dom, Byron: Mining Surprising Patterns Using Temporal Description Length. In: Gupta, Ashish (Hrsg.) ; Shmueli, Oded (Hrsg.) ; Widom, Jennifer (Hrsg.): *VLDB'98*. New York City, NY : Morgan Kaufmann, August 1998, S. 606–617

Chang u. a. 2002 Chang, Cheng-Yue ; Chen, Ming-Syan ; Lee, Chang-Hung: Mining General Temporal Association Rules for Items with Different Exhibition Periods. In: *2002 IEEE International Conference on Data Mining*. Maebashi City, Japan : IEEE Computer Society, November 2002, S. 59–66

Charikar u. a. 1997 Charikar, Moses ; Chekuri, Chandra ; Feder, Tomás ; Motwani, Rajeev: Incremental Clustering and Dynamic Information Retrieval. In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*. El Paso, Texas, USA : ACM Press, May 1997, S. 626–635

Chen und Petrounias 1999 Chen, Xiaodong ; Petrounias, Ilias: Mining Temporal Features in Association Rules. In: *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery*. Prague, Czech Republic : Springer, September 1999 (Lecture Notes in Computer Science), S. 295–300

Cheung u. a. 1996a Cheung, David W. ; Han, Jiawei ; Ng, Vincent T. ; Wong, C.Y.: Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. In: *ICDE'96*, IEEE Computer Society, 1996, S. 106–114

Cheung u. a. 1997 Cheung, David W. ; Lee, S. D. ; Kao, Benjamin: A General Incremental Technique for Maintaining Discovered Association Rules. In: *DASFAA'97*. Melbourne, Australia : World Scientific Press, April 1997, S. 185–194

Cheung u. a. 1996b Cheung, David W. ; Ng, Vincent T. ; Tam, Benjamin W.: Maintenance of Discovered Knowledge: A Case in Multi-Level Association Rules. In: *KDD'96*, 1996, S. 307–310

Das u. a. 1998 Das, Gautam ; Lin, King-Ip ; Mannila, Heikki ; Renganathan, Gopal ; Smyth, Padhraic: Rule discovery from time series.

In: *Proceedings of the Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, AAAI, August 1998, S. 16–22

Dong und Li 1999 Dong, Guozhu ; Li, Jinyan: Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In: *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego, USA : ACM Press, August 1999, S. 43–45

Ester u. a. 1998 Ester, Martin ; Kriegel, Hans-Peter ; Sander, Jörg ; Wimmer, Michael ; Xu, Xiaowei: Incremental Clustering for Mining in a Data Warehousing Environment. In: *Proceedings of the 24th International Conference on Very Large Data Bases*. New York City, New York, USA : Morgan Kaufmann, August 1998, S. 323–333

Ester und Sander 2000 Ester, Martin ; Sander, Jörg: *Knowledge Discovery in Databases: Techniken und Anwendungen*. Springer, 2000

Feldman u. a. 1997 Feldman, Ronen ; Aumann, Yonatan ; Amir, Amihood ; Mannila, Heikki: Efficient Algorithms for Discovering Frequent Sets in Incremental Databases. In: *Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'97)*. Tucson, Arizona, USA, May 1997

Frawley u. a. 1992 Frawley, William J. ; Piatetsky-Shapiro, Gregory ; Matheus, Christopher J.: Knowledge Discovery in Databases: An Overview. In: *AI Magazine* (1992), S. 57–70

Freitas 1998 Freitas, Alex A.: On Objective Measures of Rule Surprisingness. In: Zytkow, Jan M. (Hrsg.) ; Quafafou, Mohamed (Hrsg.): *Principles of Data Mining and Knowledge Discovery, Proceedings of the Second European Symposium, PKDD'98*. Nantes, France : Springer, 1998

Freitas 1999 Freitas, Alex A.: On rule interestingness measures. In: *Knowledge-Based Systems* 12 (1999), S. 309–315

Gago und Bento 1998 Gago, Pedro ; Bento, Carlos: A Metric for Selection of the Most Promising Rules. In: Zytkow, Jan M. (Hrsg.) ; Quafafou, Mohamed (Hrsg.): *Principles of Data Mining and Knowledge Discovery, Proceedings of the Second European Symposium, PKDD'98*. Nantes, France : Springer, 1998

Ganti u. a. 1999 Ganti, Venkatesh ; Gehrke, Johannes ; Ramakrishnan, Raghu: A Framework for Measuring Changes in Data Characteristics. In: *Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. Philadelphia, Pennsylvania : ACM Press, May 1999, S. 126–137

Ganti u. a. 2000 Ganti, Venkatesh ; Gehrke, Johannes ; Ramakrishnan, Raghu: DEMON: Mining and Monitoring Evolving Data. In: *Proceedings of the 15th International Conference on Data Engineering*. San Diego, California, USA : IEEE Computer Society, February 2000, S. 439–448

Grieser 2000 Grieser, Gunter: Hypothesis assessments as guidance for incremental and meta-learning. In: Keller, J. (Hrsg.) ; Girard-Carrier, C. (Hrsg.): *Proc. 11th European Conference on Machine Learning, Workshop on Meta Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, May 2000, S. 97–108

Hand und Mannila 2001 Hand, David J. ; Mannila, Heikki: *Principles of Data Mining*. The MIT Press, 2001

Hulten u. a. 2001 Hulten, Geoff ; Spencer, Laurie ; Domingos, Pedro: Mining Time-Changing Data Streams. In: Provost, Foster (Hrsg.) ; Srikant, Ramakrishnan (Hrsg.): *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York : ACM Press, August 2001, S. 97–106

Kelly u. a. 1999 Kelly, Mark G. ; Hand, David J. ; Adams, Niall M.: The Impact of Changing Populations on Classifier Performance. In: *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego : ACM Press, August 1999, S. 367–371

Kemper und Eickler 1997 Kemper, Alfons ; Eickler, André: *Datenbanksysteme – Eine Einführung*. 2nd. R. Oldenbourg Verlag, München, Wien, 1997

Keogh u. a. 2002 Keogh, Eamonn ; Lonardi, Stefano ; Chiu, Bill 'Yuan-Chi': Finding Surprising Patterns in a Time Series Database in Linear Time and Space. In: *Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. Edmonton, Alberta, Canada : ACM Press, July 2002

Klinkenberg und Joachims 2000 KLINKENBERG, Ralf ; JOACHIMS, Thorsten: Detecting Concept Drift with Support Vector Machines. In: LANGLEY, Pat (Hrsg.): *Proceedings of the 17th International Conference on Machine Learning*. Stanford, USA : Morgan Kaufmann Publishers, San Francisco, USA, 2000, S. 487–494

Lee und Cheung 1997 LEE, Sau D. ; CHEUNG, David Wai-Lok: Maintenance of Discovered Association Rules: When to update? In: *ACM-SIGMOD Workshop on Data Mining and Knowledge Discovery (DMKD-97)*. Tucson, Arizona, May 1997

Lee u. a. 1998 LEE, S.D. ; CHEUNG, D.W. ; KAO, B.: Is Sampling Useful in Data Mining? A Case in the Maintenance of Discovered Association Rules. In: *Data Mining and Knowledge Discovery* 2 (1998), September, Nr. 3, S. 233–262

Lin u. a. 2002 LIN, Jessica ; KEOGH, Eamonn ; LONARDI, Stefano ; PATEL, Pranav: Finding Motifs in Time Series. In: *Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. Edmonton, Alberta, Canada : ACM Press, July 2002

Liu u. a. 1997 LIU, Bing ; HSU, Wynne ; CHEN, Shu: Using General Impressions to Analyze Discovered Classification Rules. In: *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*. Newport Beach, USA : AAAI Press, August 1997, S. 31–36

Liu u. a. 2001a LIU, Bing ; HSU, Wynne ; MA, Yiming: Discovering the Set of Fundamental Rule Changes. In: *7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, USA, August 2001, S. 335–340

Liu u. a. 2001b LIU, Bing ; MA, Yiming ; LEE, Ronnie: Analyzing the Interestingness of Association Rules from the Temporal Dimension. In: *IEEE International Conference on Data Mining (ICDM-2001)*. Silicon Valley, USA, November 2001, S. 377–384

Mannila u. a. 1997 MANNILA, Heikki ; TOIVONEN, Hannu ; VERKAMO, A. I.: Discovery of Frequent Episodes in Event Sequences / University of Helsinki, Department of Computer Science. PO Box 26, FIN00014 University of Helsinki, Finland, February 1997 (C-1997-15). – Forschungsbericht

Masseglia u. a. 2000 Masseglia, F. ; Poncelet, P. ; Teisseire, M.: Incremental Mining of Sequential Patterns in Large Databases / LIRMM, France. January 2000. – Forschungsbericht

Morik und Rüping 2002 Morik, Katharina ; Rüping, Stefan: A Multistrategy Approach to the Classification of Phases in Business Cycles. In: *Lecture Notes in Computer Science* 2430 (2002), S. 307–??. – ISSN 0302-9743

Omiecinski und Savasere 1998 Omiecinski, Edward ; Savasere, Ashok: Efficient Mining of Association Rules in Large Databases. In: *Proceedings of the British National Conference on Databases*, 1998, S. 49–63

Park u. a. 1995 Park, Jong S. ; Chen, Ming-Syan ; Yu, Philip S.: An Effective Hash Based Algorithm for Mining Association Rules. In: Carey, Michael J. (Hrsg.) ; Schneider, Donovan A. (Hrsg.): *Proceedings of the ACM SIGMOD International Conference on Management of Data*. San Jose, California, USA : ACM Press, May 1995, S. 175–186

Parthasarathy u. a. 1999 Parthasarathy, Srinivasan ; Zaki, Mohammed J. ; Ogihara, Mitsunori ; Dwarkadas, Sandhya: Incremental and Interactive Sequence Mining. In: *Proceedings of the 1999 ACM 8th International Conference on Information and Knowledge Management*. Kansas City, USA, November 1999, S. 251–258

Patel u. a. 2002 Patel, Pranav ; Keogh, Eamonn ; Lin, Jessica ; Lonardi, Stefano: Mining Motifs in Massive Time Series Databases. In: *2002 IEEE International Conference on Data Mining*. Maebashi City, Japan : IEEE Computer Society, November 2002, S. 370–377

Pěchouček u. a. 1999 Pěchouček, Michal ; Štěpánková, Olga ; Mikšovský, Petr: Maintenance of Discovered Knowledge. In: *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery*. Prague, Czech Republic : Springer, September 1999 (Lecture Notes in Computer Science), S. 476–483

Roddick u. a. 2001 Roddick, John F. ; Hornsby, Kathleen ; Spiliopoulou, Myra: An Updated Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research. In: Roddick, John F. (Hrsg.) ; Hornsby, Kathleen (Hrsg.): *Post-Workshop Proceedings of the International Workshop on Temporal, Spatial and Spatio-Temporal Data Mining, TSDM2000* Bd. 2007. Berlin : Springer, 2001

Roddick und Spiliopoulou 1999 RODDICK, John F. ; SPILIOPOULOU, Myra: A Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research. In: *SIGKDD Explorations* 1 (1999), May, Nr. 1, S. 34–38

Savasere u. a. 1995 SAVASERE, Ashok ; OMIECINSKI, Edward ; NAVATHE, Shamkant: An Efficient Algorithm for Mining Association Rules in Large Databases. In: *Proceedings of the 21st International Conference on Very Large Data Bases*. Zurich, Switzerland : Morgan Kaufmann, September 1995, S. 432–444

Shortliffe und Buchanan 1975 SHORTLIFFE, E. ; BUCHANAN, B.: A Model for Inexact Reasoning in Medicine. In: *Mathematical Biosciences* 23 (1975), S. 351–379

Street und Kim 2001 STREET, W. N. ; KIM, YongSeog: A Streaming Ensemble Algorithm (SEA) for Large-Scale Classification. In: PROVOST, Foster (Hrsg.) ; SRIKANT, Ramakrishnan (Hrsg.): *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York : ACM Press, August 2001, S. 377–388

Syed u. a. 1999 SYED, Nadeem A. ; LIU, Huan ; SUNG, Kah K.: Handling Concept Drifts in Incremental Learning with Support Vector Machines. In: CHAUDHURI, Surajit (Hrsg.) ; MADIGAN, David (Hrsg.): *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego, USA : ACM Press, August 1999, S. 317–321. – ISBN 1-58113-143-7

Tan und Kumar 2000 TAN, Pang-Ning ; KUMAR, Vipin: Interestingness Measures for Association Patterns: A Perspective. In: *Workshop on Post Processing in Machine Learning and Data Mining at 6th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. Boston, USA, August 2000

Tan u. a. 2002 TAN, Pang-Ning ; KUMAR, Vipin ; SRIVASTAVA, Jaideep: Selecting the Right Interestingness Measure for Association Patterns. In: *Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, 2002

Thomas u. a. 1997 THOMAS, Shiby ; BODAGALA, Sreenath ; ALSABTI, Khaled ; RANKA, Sanjay: An Efficient Algorithm for the Incremental

Updation of Association Rules in Large Databases. In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD-97)*. Newport Beach, California, USA, August 1997, S. 263–266

Toivonen 1996 Toivonen, Hannu: Sampling Large Databases for Association Rules. In: *Proceedings of the 22nd Conference on Very Large Data Bases*. Mumbai (Bombay), India, September 1996

Wang u. a. 1998 Wang, Jason T. L. ; Shapiro, Bruce A. ; Shasha, Dennis ; Zhang, Kaizhong ; Currey, Kathleen M.: An Algorithm for Finding the Largest Approximately Common Substructures of Two Trees. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), August, Nr. 8, S. 889–895

Wang u. a. 1994 Wang, Jason Tsong-Li ; Chirn, Gung-Wei ; Marr, Thomas G. ; Shapiro, Bruce A. ; Shasha, Dennis ; Zhang, Kaizhong: Combinatorial Pattern Discovery for Scientific Data: Some Preliminary Results. In: Snodgrass, Richard T. (Hrsg.) ; Winslett, Marianne (Hrsg.): *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*. Minneapolis, Minnesota : ACM Press, May 1994, S. 115–125

Wang 1997 Wang, Ke: Discovering Patterns from Large and Dynamic Sequential Data. In: *Intelligent Information Systems* 9 (1997), S. 8–33

Wang und Tan 1996 Wang, Ke ; Tan, J.: Incremental Discovery of Sequential Patterns. In: *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, June 1996

Widmer und Kubat 1996 Widmer, Gerhard ; Kubat, Miroslav: Learning in the Presence of Concept Drift and Hidden Contexts. In: *Machine Learning* 23 (1996), Nr. 1, S. 69–101

Witten und Frank 1999 Witten, Ian H. ; Frank, Eibe: *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco, 1999

Zhang u. a. 2002a Zhang, Minghua ; Kao, Ben ; Cheung, David Wai-Lok ; Yip, Chi L.: Efficient Algorithms for Incremental Update of Frequent Sequences. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2002, S. 186–197

Zhang u. a. 2002b ZHANG, Minghua ; KAO, Ben ; YIP, Chi-Lap: A Comparison Study on Algorithms for Incremental Update of Frequent Sequences. In: *2002 IEEE International Conference on Data Mining*. Maebashi City, Japan, November 2002, S. 554–561

# Index